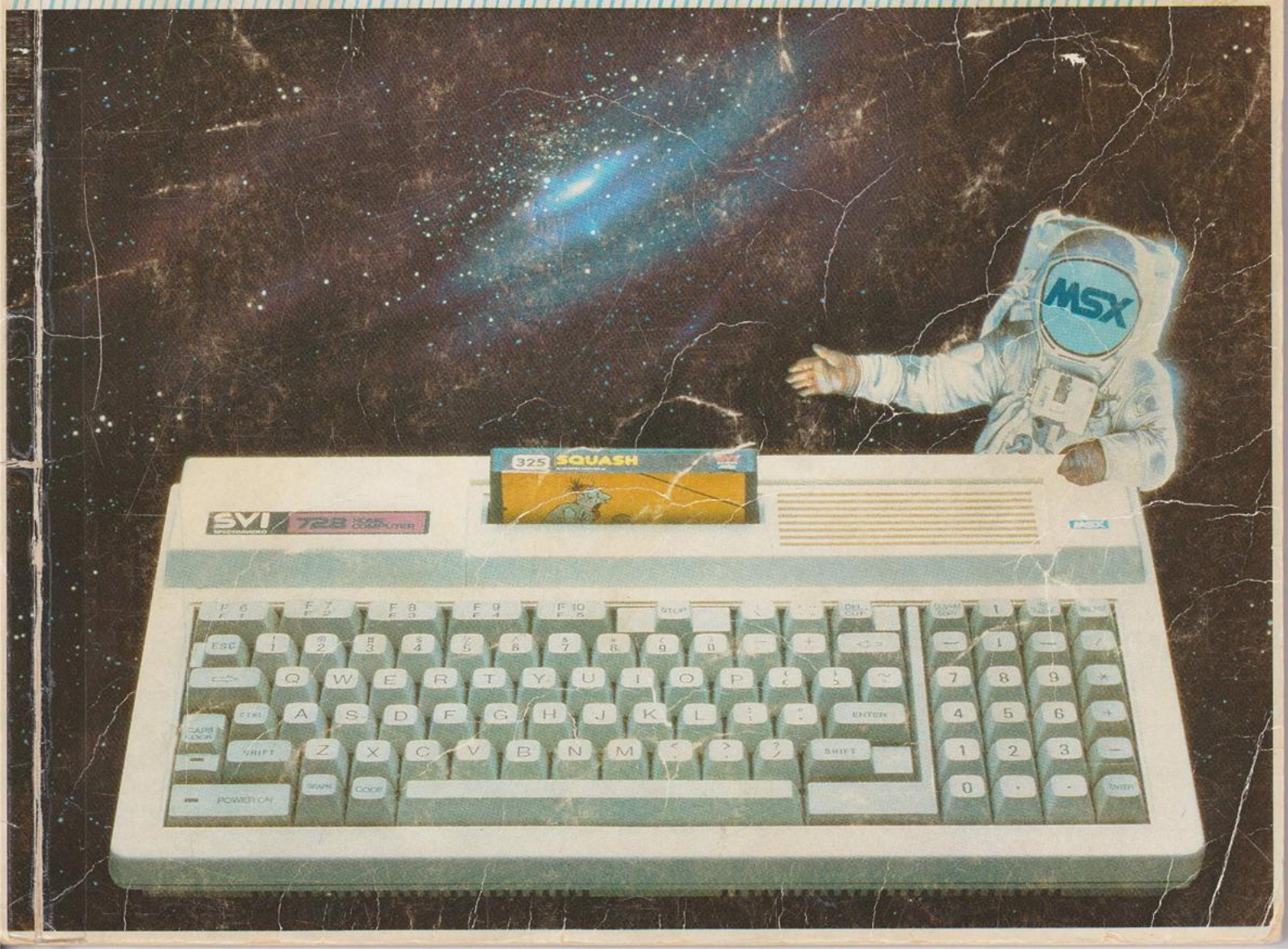
MSX Basic käskyopas



MSX
JA
SPECTRAVIDEO
KÄSKYOPAS

KUSTANTAJA

TEKNOPISTE oy
Kuusisaarentie 3 • SF-00340 HELSINKI
Puh./Tel. 90-480 011
Telex 12-2272 • Telefax 90-480 552

ISBN 951-99684-9-0
2. painos

Tämän kirjan kopioiminen, valokuvaus tai osittainenkin jäljentäminen on ilman tekijän kirjallista lupaa kielletty (tekijänoikeuslaki 404/61, muut. 897/80).

Art n:o SV-7007

Lahden Kirjapaino ja Sanomalehti Oy

SISÄLLYSLUETTELO

	AIHE	ALKAEN SIVULTA
I	JOHDANTO	5
II	YLEISTÄ MSX- JA SVI-BASIC:STÄ	6
III	TOIMINTAMUODOT	7
IV	KÄSKYN MUOTO	7
V	OHJELMARIVIN MUOTO	7
VI	MERKKIVALIKOIMA	8
VII	ERIKOISNÄPPÄIMET	8
VIII	KONTROLLIKOODIT	9
IX	VAKIOT	10
X	MUUTTUJAT	12
XI	VARATUT MUUTTUJASANAT	14
XII	TYYPPIMUUNNOKSET	15
XIII	OPERAATIOT	17
XIV	BASIC-OHJELMIEN TALTIOINTI	19
XV	TIEDON HIERARKIA	19
XVI	DATATIEDOSTOT	20
XVII	GRAFIIKKA	20
XVIII	KÄSKYJEN ESITYSTAVAN SELVITYS	21
XIX	MSX-BASIC -KÄSKYT, VÄITTÄMÄT JA FUNKTIOT	21
	BASIC-KÄSKYT (EI I/O -KÄSKYJÄ)	21
	AUTO	21
	CALL	22
	CLEAR	22
	CLICK	23
	CONT	23
	DATA	24
	DEF FN	25
	DEFINT	26
	DEFSNG	26
	DEFDBL	27
	DEFSTR	27
	DEFUSR	27
	DELETE	27
	DIM	28
	END	29
	ERASE	29
	ERROR	29
	FOR . . . NEXT	30
	GOSUB . . . RETURN	33
	GOTO	33
	IF . . . THEN . . . ELSE	34
	IF . . . GOTO . . . ELSE	35
	INPUT	35
	LET	36
	LINE INPUT	37
	LIST	37
	LLIST	38
	LPRINT	38
	LPRINT USING	38
	MID$	39
	NEW	39
	ON ERROR GOTO	40
	ON GOTO	40
	ON GOSUB	41
	PRINT	42
	PRINT USING	43
	READ	45
	REM	46
	RENUM	46
	RESTORE	47
	RESUME	47
	RUN	48
	STOP	49
	SWAP	49
	SWITCH	50
	TROFF/TRON	51
XX	ARITMEETTISET FUNKTIOT	
	ABS	51
	ATN	52
	BIN$	52
	CDBL	52
	CINT	53
	COS	53
	CSNG	53
	EXP	53
	FIX	54
	FRE	54
	HEX$	54
	INT	55
	LOG	55
	OCT	56
	RND	56
	SGN	57
	SIN	58
	SQR	58
	TAN	58
	TIME	59
XXI	MERKKIJONOFUNKTIOT	
	ASC	59
	CHR$	60
	INKEY$	61
	INPUT$	62
	INSTR	62
	LEFT$	63
	LEN	63
	MID$	63
	RIGHT$	64
	SPACE$	64
	SPC	64
	STR$	65
	STRING$	65
	TAB	65
	VAL	66
XXII	LOOGISET OPERAATTORIT	
	AND	66
	NOT	67
	OR	67
	XOR	67
	EQV	68
	IMP	68

XXIII I/O JA ERIKOISFUNKTIOT

CSRLIN 68
ERL 69
ERR 69
INP 70
LPOS 70
OUT 70
PEEK 71
POKE 72
POS 73
USR 73
VARPTR 74
WAIT 75

XXIV BASIC-VÄITTÄMÄT JA FUNKTIOT

BEEP 75
BLOAD 76
BSAVE 76
CLOAD 77
CLOAD? 77
COPY 77
CSAVE 78
KEY 78
KEY LIST 79
KEY ON/OFF 80
KILL 80
LOAD 81
MERGE 81
MOTOR 82
SAVE 82
SOUND 83

XXV KESKEYTYSPALVELUKÄSKYT

ON ERROR GOTO 83
ON INTERVAL GOSUB 84
ON KEY GOSUB 85
ON STRIG GOSUB 85
ON STOP GOSUB 86
ON SPRITE GOSUB 86
INTERVAL ON/OFF/STOP 87
KEY ON/OFF/STOP 88
STRIG ON/OFF/STOP 89
STOP ON/OFF/STOP 89
SPRITE ON/OFF/STOP 90

XXVI MUSIIKKIKÄSKYT 90

PLAY 90
SOUND 93

XXVII GRAFIIKKAKÄSKYT 95

BASE 96
CIRCLE 97
CLS 98
COLOR 98
DRAW 99
GET 101
LINE 102
LOCATE 103
PAD 104
PAINT 105
PDL 106
POINT 106
PRESET 107
PSET 107
PUT 108
PUT SPRITE 109
SCREEN 110
SPRITE$ 112
STICK 112
STRIG 113
VDP 114
VPEEK 114
VPOKE 115
WIDTH 115

XXVIII TIEDOSTOKÄSKYT 116

ATTR$ 116
CLOSE 116
COPY 116
CVI, CVS, CVD 117
DSKF 118
DSKI$ 118
DSKO$ 119
EOF 119
FIELD 119
FILES 120
FPOS 121
GET 121
INPUT# 121
INPUT$ 122
IPL 122
LFILES 123
LINE INPUT# 123
LOC 124
LOF 124
LSET 124
MAXFILES 125
MKI$, MKS$, MKD$ 125
NAME 126
OPEN 126
PRINT# 128
PRINT # USING 128
PUT 128
RSET 129
SET 130
VARPTR 130

JOHDANTO

Tämä kirja on tarkoitettu MSX- ja SPECTRAVIDEO- (myöhemmin lyhennetty SVI:ksi) BASIC:n käyttäjille. Minkä tahansa version BASIC-kielestä aikaisempi tuntemus on luonnollisesti avuksi. Kirjan tarkoituksena on helpottaa MSX BASIC -käskyjen ymmärtämistä ja antaa vihjeitä ohjelmoijalle eri tilanteissa.
Kirjassa käsitellään kaikki MSX- ja SVI-BASIC -käskyt ohjelmaesimerkein ja mukana on myös vinkkeja konekieliohjelmoijalle. Erot MSX- ja SVI-BASIC:n välillä on selvitetty jokaisen käskyn kohdalla erikseen.
Suosittelen kirjaa käytettävän yhdessä koneen kanssa, jolloin esimerkkiohjelmat ovat parhaiten hyödynnettävissä. Lisäksi kevyt silmäys kirjan sisältöön kokonaisuutena kannattaa tehdä heti alussa. Mikäli Sinulle kaikesta huolimatta jää epäselvyyksiä, ota yhteys laitteistotoimittajaasi.

Helsingissä 15.2.1985

Harri Yli-Kujala

YLEISTÄ MSX JA SVI BASIC:STÄ

BASIC on yksi monista tietokoneille kehitetyistä ohjelmointikielistä. Tänä päivänä BASIC on eniten kotitietokoneissa käytetty kieli ja siitä löytyykin useita eri versioita. MSX-BASIC on Microsoft:in kehittämän standardi BASIC-tulkin (version 4.5) laajennettu versio, joka sisältää valmiit käskyt mm. grafiikan, musiikin ja useiden oheislaitteiden ohjaukseen.
Pohjimmiltaan MSX-BASIC on luotu nk. GW-BASIC:n pohjalta, joka taas on 16-bittisille koneille kehitetty BASIC-kieli. Tärkeimpänä ajatuksena MSX-BASIC:iä kehitettäessä on ollut tehdä koko järjestelmä mahdollisimman joustavaksi ja laajennuskykyiseksi.
Myös monet ohjelmat, jotka ovat jollakin Microsoft:in BASIC-versiolla tehtyjä, ovat lähes ilman mitään muutoksia hyödynnettävissä MSX-koneilla. Eroja eri versioiden kesken esiintyy yleensä tiedostojen käsittelyyn sekä grafiikkaan kohdistuvissa käskyissä. Muuttamalla ohjelmia näiltä osin voidaan MSX-koneille sovittaa monia BASIC-ohjelmia pienellä vaivannäöllä.

(BASIC = Beginner's All purpose Symbolic Instruction Code)
(Microsoft = maailman johtava ohjelmistotalo Yhdysvalloissa)

TOIMINTAMUODOT

Kun kytket virran koneeseesi ja ROM-muistissa oleva BASIC-tulkki hetken kuluttua tulostaa kuvaruutuun tunnuksen 'Ok', olet nk. komentotasolla. Tällöin tietokoneesi on valmiina ottamaan vastaan sen ymmärtämiä käskyjä. Tästä edespäin toiminta jatkuu joko antamalla koneelle käskyjä, jotka se välittömästi suorittaa tai ohjelman muistiin kirjoituksella, ohjelmoinnilla eli epäsuorana toimintamuotona.
Lähes kaikkia käskyjä voidaan käyttää välittömässä toimintamuodossa, jolloin heti käskyn saatuaan kone sen suorittaa. Tästä esimerkki:

(kirjoita)

PRINT 5*6 (ja paina <ENTER>)
30
Ok

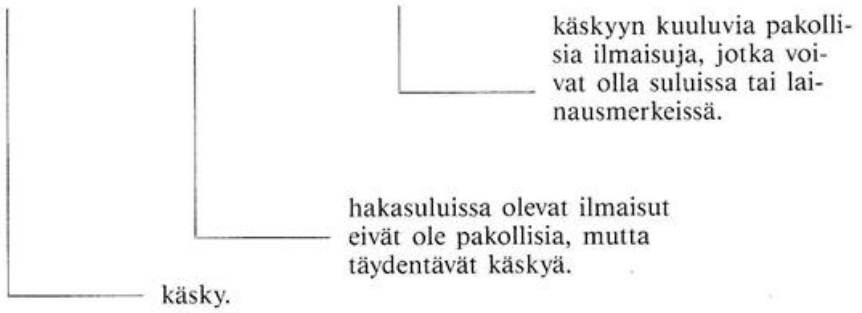
Kuvaruudulle tulee näkyviin lukujen 5 ja 6 tulo (30) ja heti tämän alle komentotilan tunnus 'Ok'. Kyseessä oli siis suora toimintamuoto, jolloin kone suoritti annetun käskyn ENTER-näppäimen painamisen jälkeen. Tässä toimintamuodossa konetta voidaan käyttää esim. laskimena.
Toinen toimintamuodoista on epäsuora toimintamuoto eli ohjelmointi. Ohjelmoinnilla pyritään jonkin tehtävän tai ongelman ratkaisuun tietokoneen avulla. Ennen kuin tietokone voi ratkaista yhtään mitään, täytyy tehtävä saattaa tietokoneen ymmärtämään muotoon ja tallentaa se koneen muistiin (RAM-muistiin) ohjelman ajoa varten. Epäsuoraa toimintamuotoa käytetään siis ohjelmien, tarkemmin ohjelmarivien taltioimiseen koneen muistiin. BASIC-ohjelma koostuu ohjelmariveistä, joilla kullakin voi olla yksi tai useampia koneen ymmärtämiä käskyjä niihin kuuluvine ilmaisuineen. Jokaista ohjelmariviä edeltää rivinumero, joka MSX- ja SVI-BASIC:ssä voi olla mikä tahansa kokonaisluku väliltä 0—65529.

KÄSKYN MUOTO

Seuraavaksi käsittelemme BASIC-käskyn syntaksin eli kieliopin. Käskystä voidaan erottaa kolme erillistä osaa: käsky, käskyyn liittyvät pakolliset ilmaisut ja käskyyn liittyvät vaihtoehtoiset ilmaisut.

ESIM:

LINE INPUT [**"<teksti>";**] **<muuttujaluettelo>**



OHJELMARIVIN MUOTO

MSX- ja SVI-BASIC:llä, kuten lähes kaikilla muillakin BASIC-murteilla on seuraavanlainen rivin formaatti (muoto):

150 PRINT "Tämäpä hauskaa!!"

Jokainen ohjelmarivi alkaa rivinumerolla, joka on kokonaisluku väliltä 0—65529. Yksi ohjelmarivi voi sisältää useita käskyjä, jolloin käskyt erotetaan toisistaan kaksoispisteellä. Yhden rivin maksimipituus on 255 merkkiä. BASIC sallii ohjelmarivien muistiin kirjoituksen mielivaltaisessa numerojärjestyksessä, sillä jokaisen rivin lisäyksen jälkeen järjestää BASIC-tulkki rivit numerojärjestykseen riviä edeltävän rivinumeron perusteella.
Ohjelmarivit voidaan myöhemmin myös numeroida uudelleen ja/tai osa riveistä voidaan poistaa tai siirtää toiseen paikkaan. Näistä toiminnoista myöhemmin RENUM- ja DELETE -käskyjen yhteydessä.

MERKKIVALIKOIMA

MSX- ja SVI-BASIC:n käyttämät merkit koostuvat aakkosista, numeroista, erikoismerkeistä ja grafiikkamerkeistä. Aakkosiin hyväksytään sekä isot että pienet kirjaimet. (HUOM! myös käskyt voidaan kirjoittaa isoilla tai pienillä kirjaimilla tai käyttäen molempia sekaisin, sillä kone muuttaa pienet kirjaimet automaattisesti isoiksi kirjaimiksi rivinsyötön yhteydessä.)
Numeroina käytetään normaaliin tapaan numeroita 0—9.
Grafiikkamerkit löydät parhaiten koneen mukana seuraavasta käyttöoppaasta.

Erikoismerkkeihin kuuluu seuraavat merkit:

MERKKI	MERKITYS
	Tyhjä eli välilyönti
=	Yhtäsuuruusmerkki
+	Plus
—	Miinus
*	Tähti tai kertomerkki
/	Kauttaviiva tai jakolasku
^ tai Ü	Eksponentti
(	Vasen sulkumerkki
)	Oikea sulkumerkki
%	Prosenttimerkki
#	Numeroa kuvaava (joskus myös punnanmerkillä ilmoitettu)
$	Dollarinmerkki (merkkijonomuuttujan tunnus)
!	Huutomerkki
[	Vasen hakasulku
]	Oikea hakasulku
,	Pilkku
.	Piste tai desimaalipiste (HUOM!)
:	Kaksoispiste

MERKKI	MERKITYS
'	Heittomerkki (saadaan SVI-328MK II:lla painamalla samanaikaisesti SHIFT+CTRL+Ö).
;	Puolipiste
&	"et"-merkki
?	Kysymysmerkki
<	Pienempi kuin -merkki
>	Suurempi kuin -merkki
\	Kokonaisluvun jakomerkki
@	"At sign" -merkki
_	Alleviivaus
<rubout>	Viimeksi kirjoitetun merkin poisto
<escape>	"Escape", jota käytetään yleensä senhetkisen toiminnan keskeyttämiseen
<tab>	Siirtää kohdistimen (kursorin) seuraavan ns. tabulointipisteen kohdalle (tabulointiväli on 8 saraketta SVI-BASIC:ssä ja 20 saraketta MSX-BASIC:ssä).
<line feed>	Siirtää kohdistimen seuraavalle riville
<ENTER>	Rivinsyöttönäppäin

ERIKOISNÄPPÄIMET

MSX-koneissa on aakkosnumeerisen näppäimistön lisäksi muutamia erikoisnäppäimiä (näppäimistön osalta koneissa on eroavaisuuksia, kuten esim. SPECTRAVIDEO:n SV-728:ssa oma erillinen numeronäppäimistö), jotka seuraavaksi selvitettäköön:

NÄPPÄIN	TARKOITUS
SHIFT	; pitämällä tätä näppäintä alaspainettuna ja painamalla samanaikaisesti haluttua toista näppäintä saadaan aikaan mm. isot kirjaimet.
CAPS LOCK	; isojen kirjainten jatkuva lukitusnäppäin (ON/OFF -tyyppinen).
CTRL	; erikoisnäppäin, jota alaspainettuna samanaikaisesti toisen näppäimen kanssa voidaan toteuttaa ns. kontrollikomentoja. Katso seuraava luku.
ESC	; "escape"-näppäin, jolla ei ole normaalisti mitään välitöntä vaikutusta. Näppäintä käytetään kuitenkin usein ohjelman keskeytykseen tai ohjelman uudelleen käynnistämiseen pysäytystilanteen jälkeen (CHR$-koodi=27).
GRAPH	; näppäin, joka alaspainettuna samanaikaisesti jonkin aakkosnumeerisen näppäimen kanssa tulostaa kuvaruutuun grafiikkamerkin. Nämä merkit löydät koneesi mukana seuraavasta käyttöohjeesta.
CODE	; pitämällä tätä näppäintä alaspainettuna ja painamalla samanaikaisesti haluttua aakkosnumeerista näppäintä, saat käyttöösi joukon erikoismerkkejä, jotka löydät koneesi mukana seuraavasta käyttöohjeesta.
F 1 — F 5 **F 6 — F 10**	; funktionäppäimet, joille on valmiiksi asetettu merkitykset, mutta jotka voidaan myös jälkeenpäin helposti uudelleen ohjelmoida (kts. KEY-käsky).
DEL **CUT**	; editointi- eli rivinmuokkausnäppäin, jota painamalla poistuu kursorin kohdalla oleva merkki.
INS **PASTE**	; editointinäppäin, jota painamalla (yhden kerran) muuttuu kursori puolikkaaksi ja kone siirtyy rivillä lisäystilaan. Kaikki merkit, jotka tämän jälkeen kirjoitetaan (seuraavaan INS- tai ENTER-näppäimen painallukseen asti), tulostuu kurso-

NÄPPÄIN	TARKOITUS
	ria ennen ja rivillä kursorin jäljessä oleva teksti siirtyy samanaikaisesti oikealle. Tämän, samoin kuin edellisen näppäimen käyttöä kannattaa harjoitella.
CLS/HM COPY	; tällä näppäimellä siirretään kursori eli kohdistin nopeasti vasempaan yläkulmaan tai painettaessa samanaikaisesti SHIFT-näppäintä, puhdistuu kuvaruutu lisäksi kaikesta tekstistä. Huomioi, että SVI-318/328:ssa toiminta on päinvastainen.
STOP	; tällä näppäimellä voidaan mm. pysäyttää kuvaruudulla juokseva ohjelman listaus. Toiminta jatkuu painamalla näppäintä uudestaan.
CTRL + STOP	; tätä yhdistelmää käytetään ohjelman keskeyttämiseen ja toiminnan siirtämiseen takaisin komentotasolle. Mikäli näppäimiä painettiin ohjelman suorituksen aikana, tulostuu kuvaruudulle teksti: Break in XXXX Ok jossa XXXX tarkoittaa rivinumeroa, jolla ohjelman suoritus oli keskeytyksen tapahtuessa. Suoritusta voidaan jatkaa keskeytyneestä kohdasta antamalla CONT-käsky.
SELECT	; tällä näppäimellä ei ole BASIC:ssä mitään merkitystä, mutta se voidaan konekielisesti ohjelmoida. Mm. grafiikkapiirtolevyä (SVI-105) käytettäessä on näppäimellä valikoiva merkitys.
ENTER	; rivinsyöttönäppäin. Tällä näppäimellä syötetään kirjoitettu käskyrivi tai esim. ohjelmassa pyydettävä tietue koneelle, joka alkaa rivin käsittelyn välittömästi tämän jälkeen. (HUOM! tämä näppäin ei vastaa kirjoituskoneen <RETURN>-näppäintä.)

KONTROLLIKOODIT

Seuraavaksi käymme läpi nk. kontrollikoodit. Näillä koodeilla tarkoitetaan välittömästi suoritettavia komentoja, jotka voidaan syöttää koneelle CTRL-näppäimen avulla. Syöttö tapahtuu pitämällä CTRL-näppäintä alaspainettuna ja painamalla samanaikaisesti jotain seuraavista näppäimistä:

NÄPPÄIN	HEKSAKOODI	ERIK. NÄPPÄIN	TOIMINTA
B *	02		; siirtää kursorin edellisen sanan alkuun.
C *	03		; "Break", jos ohjelma odottaa tiedon syöttöä.
E *	05		; tyhjentää rivin kursorin senhetkisestä kohdasta lähtien rivin loppuun asti.
F *	06		; siirtää kursorin seuraavan sanan alkuun.
G *	07		; "BEEP"-ääni.
H	08	←	; siirtää kursoria yhden sarakkeen vasemmalle poistaen samalla edellisen merkin.
I	09	→	; siirtää kursorin seuraavaan tabulointisarakkeeseen.
J *	0A		; siirtää kursoria yhden rivin alaspäin.
K *	0B	CLS/HM	; siirtää kursorin vasempaan yläkulmaan.
L *	0C	SHIFT + CLS/HM	; "Clear Screen" eli tyhjentää kuvaruudun.
M *	0D	ENTER	; vastaa <enter>-näppäimen painamista.
N *	0E		; siirtää kursorin rivin loppuun.
R *	12	INS	; lisäys/päällekirjoitus -moodin vaihtonäppäin.
U *	15		; tyhjentää rivin, jolla kursori sillä hetkellä on ja siirtää sen jälkeen kursorin rivin alkuun.
X *	18	SELECT	; ei vaikutusta BASIC:ssä SVI-328 CTRL-X ja SELECT eivät vastaa toisiaan.
	1B	ESC	; ei vaikutusta normaalissa komentotilassa.
	1C *	→	; siirtää kursoria yhden sarakkeen oikealle.
	1D *	←	; siirtää kursoria yhden sarakkeen vasemmalle.
^	1E *	↑	; siirtää kursoria yhden rivin ylöspäin.
_ *	1F	↓	; siirtää kursoria yhden rivin alaspäin.
DEL	7F	DEL	; poistaa merkin kursorin kohdalta ja siirtää rivillä olevaa tekstiä yhden sarakkeen vasemmalle.

Kahden edellisen kontrollikoodin merkitykset ovat SVI 318/328:ssa päinvastaiset.

(HUOM! tähdellä (*) merkityt koodit keskeyttävät INS-tilan ja aiheuttavat paluun komentotilaan!)

Kuten varmasti huomaat, suurin osa komennoista kohdistuu tavalla tai toisella kursoriin eli kohdistimeen. Kontrollikomentoja käytetään komentotilassa ja niiden tarkoituksena on helpottaa ohjelmarivien editointia. MSX- ja SVI-BASIC sisältää kokokuvaruutueditoinnin, jota varmasti kannattaa harjaannuttaa, sillä ohjelmien kirjoitus sujuu silloin moninkerroin nopeammin. Ohjeita editoinnin harjoittelemiseen löydät varmastikin ohjekirjasesta, joka seuraa koneesi mukana.
Lopuksi vielä lyhyt esimerkkiohjelma, jolla voit tutkia mitä koodeja eri näppäimet antavat. Ohjelma pysähtyy painamalla CTRL+STOP -näppäimiä.

```
10  A$=INPUT$(1)
20  PRINT ASC(A$),HEX(ASC(A$))
30  PRINT
40  GOTO 10
```

Rivillä 10 annetaan koneelle käsky jäädä odottamaan näppäimistöltä yhden näppäimen painallusta. Kun jotain näppäintä sitten painetaan (HUOM! SHIFT ja CTRL eivät vaikuta!), siirrytään riville 20, jolla tulostetaan painetun näppäimen ASCII-koodi ja seuraavaan tabulointikohtaan vastaava heksakoodi. Rivillä 30 tulostetaan yksi tyhjä rivi selvyyden vuoksi ja rivillä 40 ohjelmasta tehdään ikuinen silmukka käskemällä konetta aloittamaan ohjelma alusta riviltä 10. Näin jatkuu kunnes CTRL + STOP -näppäimiä painetaan. Huomioi myös STOP-näppäimen vaikutus, joka ei varsinaisesti näy.

VAKIOT

Vakioilla tarkoitetaan arvoja, joita BASIC käyttää suorituksen aikana. Yleensä vakiot määritellään heti ohjelman alussa ja niitä on kahta eri tyyppiä:

A. NUMEERISET VAKIOT

Numeeriset vakiot jaetaan kuuteen (6) eri ryhmään. Näin siksi, että esim. laskutoimituksissa, joissa ei tarvita kuin kokonaislukuja, käytetään vain kokonaislukuja (matemaattiset toiminnot kokonaisluvuilla ovat nopeampia, kuin liukuluvuilla). Myöskin kokonaislukujen varaama muistitila on pienempi, kuin muiden lukujen.
(HUOM! desimaalipisteen sijasta MSX-BASIC:ssä (kuten muissakin BASIC:ssä) käytetään desimaalipistettä. BASIC-kielessä pilkku on varattu erottamaan kaksi muuttujaa toisistaan.).
Numeeriset vakiot ovat jaetut ryhmiin seuraavasti:

1. KOKONAISLUVUT
Kokonaisluvut väliltä —32768 — +32767.

2. REAALILUVUT
Positiiviset ja negatiiviset reaaliluvut eli luvut, joissa on desimaalipiste.

3. EXPONENTTILUVUT
Luvut, jotka esitetään eksponenttimuodossa.
Eksponenttiluvut voivat olla kokonais- tai desimaaliosalla varustettuja lukuja. Eksponenttiluvut esitetään kymmenen (10) eksponentteina ja niiden lukualue on:
$1*10^{-64}$ — $1*10^{+63}$
Eksponenttiluvut merkitään vastaavasti:
1E—64 — 1 E + 63
Ja jos kyseessä on eksponenttiluku, jossa on myös desimaaliosa (kantaluku on reaaliluku), merkitään E:tä usein D:llä.

4. HEKSADESIMAALILUVUT
Heksadesimaaliluvut ovat 16-kantaisen lukujärjestelmän lukuja, joiden lukualue on:
—32768 — +65535
Vastaavat heksaluvut ovat:
&H8000 — &HFFFF
Heksadesimaalilukuja edeltää siis merkintä: &H.
Muutama esimerkki lukujärjestelmien välisistä yhtäläisyyksistä:

10-järjestelmässä	16-kantaisessa järjes.
0	&H0000
1	&H0001
10	&H000A
. . .	
100	&H0064
1000	&H03E8
1024 = 2^10	&H0400
. . .	
—1	&HFFFF
—10	&HFFF6

jne.

5. OKTAALILUVUT
Oktaaliluvut ovat kahdeksankantaisen järjestelmän lukuja, joiden lukualue on:
—32768 — +65535
Vastaavat oktaaliluvut ovat:
&O100000 — &O177777
Oktaalilukuja edeltää merkintä: &O.

6. BINAARILUVUT
Binaariluvut ovat kaksikantaisen järjestelmän lukuja, joiden lukualue on:
—32768 — +65535
Vastaavat binaariluvut ovat:
&B1000000000000000 — &B1111111111111111
Binaarilukuja edeltää merkintä: &B.
Muutama esimerkki lukujärjestelmien yhtäläisyyksistä:

BINAARILUKU	DESIMAALILUKU
&B11111000000	1984
&B10	2

Lukujärjestelmien välisiä muunnoksia voidaan myös eri järjestelmien välillä suorittaa vaivatta. Tästä myöhemmin lisää merkkijonokäskyjen yhteydessä.

YKSINKERTAISEN JA KAKSINKERTAISEN TARKKUUDEN VAKIOT

Numeeriset vakiot voidaan jakaa kahteen eri ryhmään niiden tarkkuuden mukaan. Yksinkertaisen tarkkuuden vakiot tulostetaan kuudella (6) numerolla ja ne taltiotuvat muistiin kuudella numerolla. Kaksinkertaisen tarkkuuden vakiot tulostetaan (samoin kuin taltioidaan) 14 numerolla.
Mikäli haluat käyttää konettasi laskimena, muista, että oletusarvona on kaksinkertainen tarkkuus (14 numeroa).
Yksinkertaisen tarkkuuden vakiot merkitään seuraavasti:
a. huutomerkillä (!)
esim. 777.12! 3.14!
b. eksponenttimerkinnällä (E)
esim. 707E5 —3.14E2
Kaksinkertaisen tarkkuuden vakiot merkitään seuraavasti:
a. luvun jäljessä olevalla #-merkillä
esim. 3.1415# —0.1212#
b. eksponenttimerkinnällä (D)
esim. 3.14D3 —12D2
c. jättämällä luvun perästä kaikki mahdolliset merkit pois
esim. 459 3216548 321.0045454

B. MERKKIJONOVAKIOT

Merkkijonovakio on merkkijono, joka voi sisältää kaikkia aakkosnumeerisia merkkejä ja jonka maksimi pituus on 254 merkkiä.
HUOM! JOS MSX- TAI SVI-BASIC:SSÄ KÄYTETÄÄN MERKKIJONOA, JONKA PITUUS ON YLI 200 MERKKIÄ, TULEE VASTAAVALLE MERKKIJONOLLE VARATA (TEHDÄ) TILAA CLEAR-KÄSKYLLÄ. NÄIN TULEE MENETELLÄ JOKAISEN YLI 200 MERKKIÄ PITKÄN MUUTTUJAN KANSSA.
Mikäli yrität syöttää merkkijonolle yli 254 merkkiä, ei ylimääräisiä merkkejä oteta huomioon.
Huomioi, että jos olet varannut merkkijonolle CLEAR-käskyllä tilaa 254 merkkiä ja yrität syöttää esim. INPUT-lauseella yli 254 merkkiä, EI tätä katsota virheeksi (virheilmoitusta ei tulostu kuvaruutuun ja suoritus ei siirry komentotasolle), vaan ylimääräiset merkit (yli 254) jää pois. Jos taas olet varannut tilaa esim. 240 merkkiä ja yrität syöttää vaikkapa 241, tulostuu heti virheilmoitus ja kone siirtyy komentotilaan!
Merkkijonoista ja niihin liittyvistä operaatioista (käskyistä) lisää merkkijonokäskyjen yhteydessä.

MUUTTUJAT

BASIC-kielessä muuttujilla tarkoitetaan nimiä, joita eri arvoille annetaan. Muuttujat jaetaan kolmeen ryhmään:

A. merkkijonomuuttujat

B. numeeriset muuttujat

C. taulukkomuuttujat

Nimellä osoitetaan tiettyyn muuttujaan, jonka sisältö on joko numeerinen tai aakkosnumeerinen. Numeeristen muuttujien oletusarvo on aina nolla (0) ja sekä merkkijono- että taulukkomuuttujien oletusarvo on tyhjä (nolla merkkiä).
Muuttujien nimissä hyväksytään kaikki aakkoset A—Z ja a—z sekä numerot. Kuitenkin muuttujanimen ensimmäisen merkin tulee olla kirjain. Muuttujanimissä (numeerisilla muuttujilla) sallitaan myös seuraavat erikoismerkit: % ! #.
MSX- ja SVI-BASIC:ssä voi muuttujanimi olla minkä pituinen tahansa (kuitenkin rivin maksimi pituus on 255 merkkiä), tosin vain kaksi ensimmäistä merkkiä ovat merkitseviä! Toisin sanoen esim. seuraavia kahta muuttujaa kone ei erota toisistaan:

PSTPAIK$

PSTADD$

vaan myöhemmin talletettu muuttujan arvo jää voimaan. Tästä esimerkki:

```
10  PRINT "ANNA POSTITOIMIPAIKKA"
20  INPUT PSTPAIK$
30  PRINT "ANNA POSTIOSOITE"
40  INPUT PSTADD$
50  PRINT "POSTITOIMIPAIKKASI ON ";PSTPAIK$
60  PRINT "JA POSTIOSOITTEESI ON ";PSTADD$
70  END
```

Kokeile ohjelmaa kirjoittamalla se ensin muistiin ja anna sitten RUN-käsky. Esim.

```
RUN <ENTER>
ANNA POSTITOIMIPAIKKA
? HELSINKI <ENTER<
ANNA POSTIOSOITE
? VAASANKATU 2 <ENTER>
POSTITOIMIPAIKKASI ON VAASANKATU 2     <- - - - HUOM!
JA POSTIOSOITTEESI ON VAASANKATU 2     <- - - - HUOM!
Ok
```

Koska MSX BASIC tunnistaa muuttujat vain kahden ensimmäisen merkin perusteella, tulostuu esimerkkiohjelmassa sekä PSTPAIK$:n että PSTADD$:n kohdalla viimeksi annettu tieto (VAASANKATU 2).
Muuttujanimi ei voi olla mikään BASIC:n käskysanoista, väittämistä, operaatio- eikä funktionimistä. Muuttujanimi ei voi myöskään alkaa kirjainyhdistelmällä FN, koska silloin lauseen oletetaan olevan käyttäjän määriteltävissä oleva oma funktio.
Huom! MSX-BASIC:ssä voidaan muuttujanimen toiseksi kirjaimeksi antaa grafiikkamerkki ja näin laajentaa muuttujanimien määrää huomattavasti.

A. MERKKIJONOMUUTTUJAT

Merkkijonomuuttujat merkitään $-merkillä, joka lisätään muuttujanimen perään. Muuttujan maksimi pituus on 254 merkkiä (HUOM! yli 200 merkkiä pitkille muuttujille tulee varata tilaa CLEAR-käskyllä) ja merkkijonomuuttuja voi olla myös nk. indeksoitumuuttuja.
Indeksoidulla muuttujalla tarkoitetaan muuttujaa, jossa on nimen lisäksi indeksiosa, jonka perusteella kone muuttujan muistiinsa taltioi. Indeksi on aina kokonaisluku, mutta voi olla myös itse muuttuja.
Esimerkkejä merkkijonomuuttujista:

A$ NIMI$ ADDR$ TAULU$(12) KARD$(A)

Kaksi viimeistä esimerkkiä ovat indeksoituja muuttujia, joista edellisessä indeksinä on luku 12 ja jälkimmäisessä indeksiksi tulee muuttujan A sen hetkinen arvo. Mikäli muuttuja A ei ole kokonaisluku, suoritetaan pyöristys lähimpään pienempään kokonaislukuun.
Kun merkkijonomuuttujille annetaan jokin merkkijono, täytyy merkkijono merkitä lainausmerkkeihin. Esim.

A$="TIETOKONE" NIMI$="PEKKA" TAULU$(12)="Kerroin 154"

Merkkijono voi sisältää siis myös välilyöntejä, numeroita, isoja ja pieniä kirjaimia. Merkkijonojen varaama muistitila on kolme (3) tavua + yksi (1) tavu jokaista merkkijonon merkkiä kohti (myös välilyöntiä eli tyhjää).

B. NUMEERISET MUUTTUJAT

Numeerisia muuttujia on kolmea (3) eri tyyppiä:

1. KOKONAISLUKUMUUTTUJAT

Kokonaislukumuuttujien jäljessä on aina %-merkki kertomassa koneelle, että kyseessä todellakin on kokonaisluku (oletusarvona kaksinkertaisen tarkkuuden luku).
Kaikissa matemaattisissa yhtälöissä, FOR-NEXT -silmukoissa ja muissa tapauksissa, kun vain mahdollista, kannattaa käyttää kokonaislukuja, sillä niiden muistista varaama tila on vain kaksi (2) tavua ja ohjelman suoritus nopeutuu huomattavasti (lähes puolella suurissa silmukoissa).
Esimerkkejä kokonaislukumuuttujista:

A% KAT% DATA%(KIER%)

Viimeisin esimerkki on indeksoitu kokonaislukumuuttuja.

2. YKSINKERTAISEN TARKKUUDEN MUUTTUJAT

Yksinkertaisen tarkkuuden muuttujien nimen jäljessä on !-merkki ja ne voivat myös olla indeksoituja. Ko. muuttuja varaa muistia neljä (4) tavua.
Esimerkkejä:

A! MAKSIMI! TAULU!(1)

3. KAKSINKERTAISEN TARKKUUDEN MUUTTUJAT

Kaksinkertaisen tarkkuuden muuttujan nimen jäljessä on #-merkki ja myös indeksoidut muuttujat hyväksytään. Ko. muuttuja varaa jokaista muuttujaa kohti kahdeksan (8) tavua tilaa.
Esimerkkejä:

PI# TUPLA# AIKA#(IND)

C. TAULUKKOMUUTTUJAT

Taulukkomuuttujilla eli indeksoiduilla muuttujilla tarkoitetaan muuttujia, joilla on sama muuttujanimi, mutta joissa on muuttujanimen lisäksi suluissa oleva indeksiosa. Tätä indeksiosaa kutsutaan usein alkioksi. Esim:

MIN(5) A$(23) B(12,8) TEST#(5,4,8)

Kaksi ensimmäistä esimerkkiä ovat yksiulotteisen taulukon alkioita, seuraava alkio on kaksiulotteisesta taulukosta ja viimeinen esimerkki on kolmiulotteisesta taulukosta.
Käytettäessä taulukoita, joissa on alkioita yhdessä suunnassa enemmän kuin 11, tulee tällaiselle taulukolle varata tilaa ennalta (usein ohjelman alussa) DIM-käskyllä (kts. DIM-käsky). Taulukon pienin alkio on aina nolla (0), sillä MSX-BASIC ei tunne OPTION BASE -käskyä, jolla mm. MBASIC:ssä voidaan pienin alkio määritellä joko nollaksi tai yhdeksi.
Taulukko muuttujineen voi olla siis yksi-, kaksi- tai kolmiulotteinen. Taulukon alkiot voivat sisältää numeerista tai aakkosnumeerista tietoa, mutta vain jompaa kumpaa. Alkiot voivat siis olla kokonaislukuja, yksinkertaisen- tai kaksinkertaisen tarkkuuden lukuja tai merkkijonoja.
Määriteltäessä taulukoita DIM-käskyllä tulee muistaa, että ne varaavat usein (varsinkin kaksi- ja kolmiulotteiset) paljon muistitilaa, joten suuria taulukoita kannattaa säilyttää aina oheismuistissa (esim. taulukon lataus levykkeeltä kestää usein vain sekunnin, pari). Taulukot on syytä mitoittaa melko niukasti, sillä niiden suurentaminen on helppoa (kts. DIM-käsky).
Taulukon koko on rajattu siten, että maksimi määrä alkioita kaikilla kolmella matriisin (taulukon) tasolla on 255. Teoreettinen matriisin maksimi koko olisi A(254, 254, 254), mutta käytännössä taulukon kokoa rajoittaa käytettävissä oleva RAM-muisti. Maksimi taulukko vaatisi keskusmuistia jopa yli 133 miljoonaa tavua tyhjänä!
Koska matriisin pienin alkio on aina nolla (0), käytettäessä esim. taulukkoa (7,9) on käytössä alkiot (0,1,2. . .,6,7) X-akselin suuntaan (yht. 8 kpl) ja alkiot (0,1,2,. . .,8,9,) Y-akselin suuntaan (yht. 10 kpl).
Mikäli käytät pienempää kuin 11-alkion matriisia (esim. taulukkoa A(8)), kannattaa tällainen taulukko ohjelman alussa tai viimeistään ennen taulukon käyttöönottoa määritellä. Matriisi A(8) varaa tilaa yhdeksälle alkioille ja koska taulukon jokaisen suunnan alkioiden määrän oletusarvona on 11 kpl (esim. A(10)), jäisi taulukosta A(8) kaksi alkiota käyttämättä. Määrittely DIM-käskyllä tällaisessa tapauksessa ei ole pakollista, mutta suositeltavaa.
Esimerkki taulukon käytöstä:

```
10  REM TAULUKKO-OHJELMA
20  CLS
30  PRINT "TÄMÄ OHJELMA TALLENTAA HALUTUN"
40  PRINT "MÄÄRÄN NIMIÄ JA PUHELINNUMEROI-"
50  PRINT "TA MUISTIIN KYSYMÄLLÄ JOKAISEN"
60  PRINT " NIMEN JA NUMERON ERIKSEEN."
70  PRINT : PRINT
80  INPUT "ANNA NIMIEN/PUH.NUM. MÄÄRÄ"; MAAR%
90  DIM NIM$(MAAR%—1)
100 DIM NUM#(MAAR%—1)
110 FOR KIER%=1 TO MAAR%
```

```
RUN <ENTER>
 3
Ok
```

Rivillä 10 annetaan kokonaislukumuuttujalle PII piin likiarvo 3.141. Koska muuttuja PII on kuitenkin merkitty kokonaislukumuuttujaksi, saa muuttuja PII arvon kolme (3), joka rivillä 20 PRINT-käskyllä tulostetaan. Huom! %-merkki on kokonaislukumuuttujan merkki.

2. Kun muuttujalle lasketaan aritmeettisen tai relaatio-operaation aikana arvoa, muutetaan kaikki luvut/operandit samaan tarkkuuteen, joka on tarkimman luvun/operandin tarkkuus. Myös aritmeettisen operaation tulos muunnetaan samaan tarkkuuteen.

Esim:

```
10   A=1/3!
20   PRINT A
30   END

RUN <ENTER>
 .33333333333333
Ok
```

Rivillä 10 oleva aritmeettinen operaatio suoritetaan käyttäen kaksinkertaista tarkkuutta ja saatu osamäärä tulostetaan käyttäen samaa tarkkuutta rivillä 20.

```
10   A!=1/3
20   PRINT A!
30   END

RUN <ENTER
 .333333
Ok
```

Edellisessä esimerkissä itse aritmeettinen operaatio suoritetaan käyttäen kaksinkertaista tarkkuutta, mutta tulos sijoitetaan yksinkertaisentarkkuuden muuttujalle A!. Rivillä 20 tulostetaan saatu osamäärä.

3. Loogiset operaattorit muuntavat operandinsa kokonaisluvuiksi ja antavat myös tuloksen aina kokonaislukuna. Operandit voivat vaihdella välillä —32768 — +32767, muuten tulostetaan virheilmoitus "Overflow" eli ylivuoto.

4. Muunnettaessa reaalilukuja kokonaisluvuiksi poistetaan muunnettavasta luvusta desimaaliosa (pyöristystä EI suoriteta!).

Esim:

```
10   A%=31.41
20   PRINT A%
30   END

RUN <ENTER>
 31
Ok
```

Rivillä 10 poistetaan luvusta 31.41 desimaaliosa ja saatu kokonaisluku talletetaan A:n arvoksi.

5. Muunnettaessa kaksinkertaisentarkkuuden muuttujaa yksinkertaiseen tarkkuuteen otetaan vain kuusi (6) ensimmäistä desimaalia mukaan. Tarvittaessa suoritetaan kuudennelle desimaalille normaali pyöristys ylöspäin.

Esim:

```
10   A!=LOG(5)
20   B=A!
30   PRINT A!;B;LOG(5)
40   END
```

Rivillä 10 saa yksinkertaisentarkkuuden muuttuja A LOG(5):n arvon. Rivillä 20 sama arvo talletetaan muuttujalle B, joka tosin on kaksinkertaisentarkkuuden muuttuja, mutta saa nyt vain kuusi desimaalia. Rivillä 30 olevan PRINT-lauseen avulla on tehty pyöristys havaittavissa.

OPERAATIOT

MSX- ja SVI-BASIC:ssä operaatiot voidaan eritellä neljään eri ryhmään:

1. RELAATIO-OPERAATIOT

2. ARITMEETTISET OPERAATIOT

3. LOOGISET OPERAATIOT

4. KÄSKYN SISÄLTÄVÄT OPERAATIOT

Lisäksi yhtenä erillisenä operaatioryhmänä näiden ulkopuolella voidaan pitää merkkijonoihin kohdistuvia operaatioita. Operaatiot kohdistuvat lukuihin joko matemaattisina tai loogisina toimenpiteinä.

1. RELAATIO -OPERAATIOT

Relaatio-operaatioilla suoritetaan aina kahden eri arvon vertailuja ja puhutaankin yleensä relaatiotesteistä. Testin tulos on joko TOSI eli "true" (—1) tai EPÄTOSI eli "false" (0). Testin tulosta käytetään yleensä IF-ehtolauseessa ohjelman suoritusjärjestyksen vaikuttavana tekijänä.

Aritmeettiset- ja relaatio-operaatiot suoritetaan aina ennen loogisia operaatioita. Ja mikäli nämä kaksi operaatiota suoritetaan yhden ilmaisun sisällä, suoritetaan aritmeettinen operaatio aina ensin. Testi voi kohdistua kahteen muuttujaan tai vakioon ja kaikki seuraavat testit ovat sallittuja:

OPERAATIO	TESTI
=	; kahden muuttujan yhtäsuuruus testi
<>	; kahden muuttujan erisuuruus testi
>	; suurempi kuin -testi
<	; pienempi kuin -testi
>=	; suurempi tai yhtäsuuri kuin -testi
<=	; pienempi tai yhtäsuuri kuin -testi

Esim:
A*B>=A*10—C

Tämä testi saa arvon TOSI, jos muuttujien A ja B tulo on suurempi tai yhtäsuuri kuin kymmenen (10) kertaa muuttujan A tulo vähennettynä muuttujan C sen hetkisellä arvolla, muuten lause saa arvon EPÄTOSI.

Seuraavaksi esimerkki relaatiotestin käytöstä BASIC-lauseessa:

IF SQR(A*3)<C+B THEN PRINT "OIKEIN !" ELSE PRINT "VÄÄRIN !"

Mikäli lauseen A*3 neliöjuuri saa pienemmän arvon kuin muuttujien C ja B summa, tulostaa ohjelmarivillä oleva PRINT-lause 'OIKEIN !' ja muussa tapauksessa 'VÄÄRIN !'.

2. ARITMEETTISET OPERAATIOT

Aritmeettisiin operaatioihin kuuluu seuraavat operaatiot (operaatiot on esitetty järjestyksessä, jossa ne suoritetaan, mikäli niitä on yhdessä lauseessa useampia):

OPERAATIO	TOIMINTA
^	; kantaluvun potenssiin korotus
—	; luvun vastaluku eli negaatio
* ja /	; lukujen tulo ja osamäärä (suoritetaan järjestyksessä vasemmalta oikealle)
+ ja —	; yhteen- ja vähennyslasku (suoritetaan vasemmalta oikealle)

Eri operaatioiden suoritusjärjestykseen voidaan vaikuttaa käyttämällä sulkuja "(" ja ")", jolloin operaatiot suoritetaan alkaen sisimmistä sulkulausekkeista. Sulkulausekkeiden sisällä on voimassa samat laskujärjestyssäännöt. Sulkujen määrää ei ole rajoitettu.

Esim:
A*B—10^C+(5+A)

Esimerkkilause suoritettaisiin seuraavassa järjestyksessä: (oletetaan, että muuttujien arvot ovat seuraavat)

A=2 B=3 C=4
1. 5+A eli 5+2, joka on 7
2. 10^C eli 10^4, joka on 10000
3. A*B eli 2*3, joka on 6
4. 6—10000+7, joka on —9987

Sulkulausekkeet suoritetaan siis aina ensimmäisinä.

MSX- ja SVI-BASIC tuntee myös kokonaisjako- ja jäännösjakofunktiot. Kokonaisjakofunktio saadaan käyttämällä "\"-merkkiä, jolloin jaettava ja jakaja muunnetaan ennen varsinaista jakolaskua kokonaisluvuiksi (HUOM! sekä jaettavan että jakajan tulee olla lukuja välillä —32768 — +32767).

Saatu osamäärä muunnetaan (mikäli se ei ole kokonaisluku) kokonaisluvuksi. Funktiota merkitään joskus myös merkinnällä DIV.

Esim:

PRINT 34.1\7.13 <ENTER> vrt. **34.1/7.13=4.7826086956522**
4
Ok

PRINT 10\3 <ENTER> vrt. **10/3=3.3333333333333**
3
Ok

Modulusaritmetiikan tunnuksena on MOD-merkintä. MOD palauttaa kokonaisluvun kahden kokonaisluvuksi muunnetun (mikäli jaettava tai jakaja eivät ole kokonaislukuja) luvun osamäärästä jäävän jakojäännöksen. Siitä nimi jäännösjakofunktio.

Esim:

PRINT 34.1 MOD 7.13 <ENTER> **vrt. 34\7 = 4 ja jäännös 6**
6 **eli 34—7=27—7=20—7=13—7= 6**
Ok

PRINT 10 MOD 3 <ENTER> **vrt. 10\ = 3 ja jäännös 1**
1 **eli 10—3=7—3=4—3= 1**
Ok

Nollalla (0) jakaminen ei ole sallittua ja jos niin tapahtuu, tulostetaan virheilmoitus "Division by zero". Tai mikäli operandien arvot ylittävät sallitut rajat, tulostuu virheilmoitus "Overflow".

3. LOOGISET OPERAATIOT

MSX- ja SVI-BASIC sisältää kaikki loogiset operaatiot:

NOT, AND, OR, XOR, EQV ja IMP.

Loogisilla operaatioilla suoritetaan vakioille ja/tai muuttujille relaatiotestejä (kahden arvon vertaus), BOOLEAN operaatioita ja bittikäsittelyä. Operaatiosta saatava tulos on joko TOSI (ei nolla) tai EPÄTOSI (nolla).
Loogiset operaatiot on esitelty jäljempänä omana lukunaan vastaavien käskyjen yhteydessä.

4. KÄSKYN SISÄLTÄVÄT OPERAATIOT ELI FUNKTIO-OPERAATIOT

MSX- ja SVI-BASIC sallii funktioiden käytön ilmaisuissa kutsumaan aikaisemmin määriteltyjä operaatioita, mutta sisältää myös valmiita funktioita kuten neliöjuuri (SQR), logaritmi (LOG), jne. Myös käyttäjän omien funktioiden käyttö on mahdollista (kts. DEF FN -käsky) yhdistämällä valmiita funktioita nk. funktiojonoiksi.

5. MERKKIJONO-OPERAATIOT

Merkkijonoihin liittyvät käskyt on esitelty omana lukuna merkkijonokäskyissä. Käsittelemme tässä merkkijonoihin liittyviä yleisiä muistisääntöjä.

A. MERKKIJONOJEN VERTAILUT

Merkkijonoja voidaan vertailla samoilla käskyillä, kuin numeerisia vakioita/muuttujia. Relaatio-operaatiot olivat seuraavat:

= <> > < >= <=

Vertailu merkkijonoilla tapahtuu vertaamalla kahden merkkijonon yhtä merkkiä (merkin ASCII-koodia) kerrallaan toisiinsa. Mikäli kummankin merkkijonon kaikkien merkkien ASCII-koodit ovat samat, ovat merkkijonot identtiset. ASCII-koodeihin liittyy muutama helppo muistisääntö. Ison kirjaimen koodiluku on vastaavaa pientä kirjainta aina edellä (pienempi koodiluku) ja merkkijonoja sekä edeltävät että seuraavat tyhjät (välilyönti) ovat merkityksellisiä, samoin kuin merkkijonojen välissä olevat tyhjät (ASCII-koodi on 32). Lisäksi lyhyempää merkkijonoa sanotaan pienemmäksi.

B. MERKKIJONOJEN YHDISTÄMINEN

Merkkijonoja voidaan myös yhdistää "+"-merkillä ja samalla muuttaa yhdistetyssä merkkijonossa esiintyvien merkkijonojen järjestystä.
(HUOM! merkkijono EI saa ylittää 254 merkkiä).
Kun merkkijonoja vertaillaan, tulee ne merkitä lainausmerkkeihin.

Esim:

"A" < "B"	**TOSI**
"1" < "2"	**TOSI**
"JUSSI" = "JUSSI"	**TOSI**
"Jussi" = "JUSSI"	**EPÄTOSI**
"Jussi" > "JUSSI"	**TOSI**
"JUSSI" > "JUSSI"	**TOSI**
"1/1/1985" < "2/1/1984"	**TOSI**
NIMI$ > "HEIKKI"	**TOSI, kun NIMI$="HILKKA"**

Myös lukuja ja grafiikkamerkkejä sisältäviä merkkijonoja voidaan vertailla.

DATATIEDOSTOT

Datatiedostot ovat siis tietoa sisältäviä muistilohkoja, joita säilytetään oheismuistissa. Tiedostoja on kahta eri tyyppiä:

1. SARJASAANTITIEDOSTOT ELI SEQUENTIAL FILES

2. HAJASAANTITIEDOSTOT ELI RANDOM ACCESS FILES

Kumpaankin tiedostotyyppiin liittyy sekä tiedostonumero että tiedostonimi.
Tiedostonumeroa käyttää tietokone operoidessaan tiedoston kanssa. Aina tiedostoon kirjoituksen tai sieltä luvun yhteydessä tietokone identifio jokaisen tiedoston omalla numerolla.
Tiedostonimeen liittyy kaksi seikkaa:

a. laite eli device, johon tiedostoa ollaan kohdistamassa
b. tiedostonimi eli filename, jolla tiedosto on esim. levykkeelle talletettu

Nämä kaksi ilmaisua erotetaan toisistaan aina kaksoispisteellä (:).
MSX- ja SVI-BASIC tuntee seuraavat "laitteet":

LAITE	VERSIOT	TARKOITTAA
CRT	;	kuvaruutu eli screen. Käytetään vain tiedon tulostukseen. Sekä MSX että SVI kasetti- ja levykeversiossa.
LPT	;	kirjoitin eli printer. Käytetään vain tiedon tulostukseen. Sekä MSX että SVI kasetti- ja levykeversiossa.
CAS	;	kasettiyksikkö eli cassette tape player. Käytetään tiedon tallennukseen ja lukuun. Sekä MSX että SVI kasetti- ja levykeversiossa.
GRP	;	tarkkuusgrafiikkatila eli graphic screen. Käytetään tiedon tallennukseen. Sekä MSX kasetti- että levykeversiossa.
1	;	levykeasema yksi (1). Käytetään tiedon lukuun ja kirjoitukseen. Vain SVI-levykeversiossa.
2	;	levykeasema kaksi (2). Sama kuin edellä.
A	;	levykeasema yksi (1). Käytetään tiedon lukuun ja kirjoitukseen. Vain MSX-levykeversiossa.
B	;	levykeasema kaksi (2). Käytetään tiedon lukuun ja kirjoitukseen. Vain MSX-levykeversiossa.
KYBD	;	näppäimistö eli keyboard. Vain tiedon luku mahdollinen. Toimii vain SVI-BASIC:ssä.

Tiedostonimi koostuu kahdesta osasta seuraavasti:

nimi . laajennusosa

Nämä kaksi ilmaisua erotetaan toisistaan pisteellä, kuten esimerkissä. Nimiosan 'nimi' maksimi pituus on kahdeksan (8) merkkiä (MSX) tai kuusi (6) merkkiä (SVI). Ilmaisu 'laajennusosa' voi sisältää kolme (3) merkkiä, ylimääräisiä ei huomioida (taltioida). Ilmaisu 'laajennusosa' ei ole pakollinen, mutta sitä käytetään yleensä kertomaan tiedoston sisällöstä. Tästä muutama tyypillinen esimerkki:

TIEDOSTONIMI		TARKOITTAA
VARAST.DATA	;	**BASIC-datatiedosto**
SINI.BAS	;	**BASIC-kielinen ohjelma**
KELLO	;	**---II---**

GRAFIIKKA

MSX- ja SVI-BASIC:n yksi parhaista puolista on sen helppokäyttöinen grafiikka. Kaikissa MSX-koneissa ja SPECTRAVIDEO:ssa käytetään grafiikan ja tekstin tuottamiseen piiriä TMS 9918/9929 Video Display Processor. Tätä samaa piiriä käytetään monissa mikrotietokoneissa sen erinomaisen kyvyn tuottaa tekstiä, erikoismerkkejä, yksittäisiä pisteitä, viivoja, käyriä ja erityisesti SPRITE-kuvahahmoja vuoksi.
Käytettävissäsi on 16 väriä ja 32 samanaikaisesti ohjattavissa olevaa SPRITE-hahmoa. Näiden käytöstä yksityiskohtaisemmin vastaavien käskyjen yhteydessä.
Tulostus kuvaruudulla on jaettu neljään (4) moodiin (MSX) tai kolmeen (3) moodiin (SVI) seuraavasti:

MSX	SVI
A. TEKSTITILA	**A. TEKSTITILA**
B. LEVENNETTY TEKSTITILA	**B. TARKKUUSGRAFIIKKATILA**
C. TARKKUUSGRAFIIKKATILA	**C. LOHKOGRAFIIKKATILA**
D. LOHKOGRAFIIKKATILA	

Tila, johon tullaan virran kytkemisen jälkeen, on tekstitila (A). Sekä tilassa A että B (MSX) voidaan suorittaa ohjelmien muistiin taltiointia. Molemmissa tiloissa taustan väri peittää koko kuvaruudun ja näkyvissä ovat myös funktionäppäinten merkitykset (ellei sitä ole estetty). Tässä tilassa kuvaruutu on jaettu 24:n riviin (X'PRESS:ssä

25:n riviin), joilla kullakin on 1—40 tai 80 merkkiä/rivi. Oletusarvona koneen kytkemisen jälkeen on rivillä 39 merkkiä, mutta käyttämällä WIDTH-komentoa voidaan rivin leveys määritellä vapaasti väliltä 1—40 (MSX). Käytettäessä 80 merkkiä/rivi -laajennuskorttia saadaan näkyville 80 merkkiä riville. Tällöin käytössä tulee olla erillinen videomonitori, sillä television erottelukyky ei riitä (poikkeuksena SVI-738 X'PRESS).
Rivi 24 (X'PRESS:ssä rivi 25) on varattuna funktionäppäimille, mutta voidaan ottaa BASIC:n käyttöön koska tahansa KEY OFF -käskyllä (MSX) tai 'SCREEN ,0' -käskyllä (SVI).
Laajennetussa tekstitilassa (MSX) näytöllä on 32 merkkiä rivillä ja myös tässä tilassa voidaan ohjelmia normaalisti editoida. Kirjainten ja merkkien väli on noin puolet merkin leveydestä.
Tarkkuusgrafiikkatilassa ei ohjelmia voida editoida, vaan se on varattu yksinomaan grafiikan tulostukseen. Näyttö on jaettu 256 pisteeseen vaakasuunnassa ja 192 pisteeseen pystysuunnassa. Jokainen kuvaruudun piste voidaan sytyttää erikseen.
Lohkografiikkatilassa on näyttö jaettu 64 pisteeseen vaakasuunnassa ja 48 pisteeseen pystysuunnassa.
Tarkkuusgrafiikassa voidaan kuvaruutu taltioida myös oheismuistiin, joko nauhalle tai levykkeelle (kts. SAVE-käsky).
Ääniefekteistä huolehtii oma prosessori ja jopa kolmiäänisen musiikin tuottamiseen on MSX- ja SVI-BASIC:ssä varattuna omat käskynsä, joista myöhemmin lisää.

KÄSKYJEN ESITYSTAVAN SELVITYS

Seuraavaksi esitellään kaikki MSX- ja SVI-BASIC:n käskyt, funktiot ja väittämät. Jokainen käsky käsitellään seuraavasti:

KÄSKY

MUOTO: ; muodolla annetaan käskyn tarkka muoto, jota tulee noudattaa. Hakasuluissa olevat ilmaisut eivät ole pakollisia.

VERSIOT: ; versiossa on selvitetty käskyn esiintyminen erikseen MSX- ja SVI-BASIC:n kasetti- (KASET.) ja levykeversiossa (LEVYKE).

TARKOITUS: ; selvittää mihin tarkoitukseen käskyä käytetään.

LISÄYKSET: ; antaa käskyyn liittyvistä ilmaisuista ja erikoisominaisuuksista tarkat lisätiedot.

ESIM: ; antaa käskyn käytöstä käytännön esimerkin.

Huomaa ero MSX- ja SVI-BASIC:n potenssimerkissä:

MSX ^
SVI ^ **(MK I)**
SVI Ü **(MK II)**

MSX-BASIC KÄSKYT, VÄITTÄMÄT JA FUNKTIOT

BASIC-KÄSKYT (EI I/O-KÄSKYJÄ)

AUTO

MUOTO: **AUTO** [<**rivinumero**>] [,<**lisäys**>]

VERSIOT: **MSX: KASET. + LEVYKE**
SVI: KASET. + LEVYKE

TARKOITUS: Tuottaa automaattisesti rivinumeroita ENTER-painikkeen painamisen jälkeen.

LISÄYKSET: <rivinumero> = ensimmäinen rivinumero, josta rivinumerointi alkaa.
<lisäys> = luku, joka lisätään automaattisesti aina ENTER-painikkeen painamisen jälkeen edelliseen rivinumeroon.
ENTER:in painamisen jälkeen tulostuu uusi rivinumero ja BASIC odottaa käskyrivin kirjoittamista. Mikäli rivinumeron perään tulostuu tähti (*), on kyseinen rivinumero jo käytössä. Painamalla silloin ENTER-painiketta säilyy entinen käskyrivi muuttumattomana ja AUTO-käsky muodostaa uuden rivinumeron.

Ilmaus <rivinumero> ja/tai <lisäys> voidaan jättää antamatta. Molempien oletusarvona on 10, paitsi jos ilmaus <lisäys> annetaan ilman ilmausta <rivinumero>, jolloin oletusarvona <rivinumero>:lle on nolla (0). Mikäli AUTO-käskyssä annetaan ilmaus <rivinumero> ja tämän perässä pilkku (,), käytetään lisäyksenä viimeksi käytettyä lisäystä.
AUTO-toiminnosta pääsee pois painamalla samanaikaisesti CTRL+STOP -painikkeita tai CTRL+C -painikkeita (Break), jonka jälkeen suoritus siirtyy komentotilaan. Editointi eli käskyrivin muokkaus on sallittua AUTO-käskyn aikana.

ESIM:

KÄSKY	**ANTAA RIVINUMEROT**	
AUTO	**10,20,30, . . .**	
AUTO 1	**1,11,21,31, . . .**	
AUTO ,2	**0,2,4,6, . . .**	**HUOM!**
AUTO 100,15	**100,115,130,145, . . .**	

AUTO-käsky (10,10) käynnistyy myös funktionäppäimestä kaksi (2). Komennolla 'AUTO.' alkaa rivinumerointi viimeksi käsitellyltä riviltä alkaen, jolloin myös ko. rivinumeron perään tulostuu tähti (*).
Uutta ohjelmaa ei yleensä kannata alkaa riviltä 10, koska ohjelman alkuun joudutaan usein lisäämään useita rivejä. Tästä syystä suositellaan ohjelman kirjoittamisen alkamista käskyllä 'AUTO 100'. Nyt ensimmäiseksi rivinumeroksi tulee 100, seuraavaksi 110, jne. Ohjelmarivien uudelleennumerointi käy helposti myöhemmin RENUM-käskyllä.
Jokainen rivinumero varaa tilaa kaksi (2) muistipaikkaa RAM-muistia huolimatta rivinumerosta, eikä näin ollen ohjelmasi varaama muistitila riipu rivinumeroista.

CALL

MUOTO: **CALL <nimi> [(<arglist>)]**

VERSIOT: **MSX: KASET. + LEVYKE**
SVI: ei käytössä

TARKOITUS: Käskyllä siirrytään konekieliseen aliohjelmaan nimeltä <nimi>.

LISÄYKSET: <nimi> = ROM-muistissa olevan ohjelman nimi (absoluuttiosoite).
<arglist> = lista aliohjelmaan välitettävistä argumenteista, jotka erotetaan lainausmerkeillä ja toisistaan pilkuilla.
CALL-käskysana voidaan korvata merkillä "_".

ESIM: **CALL FORMAT <ENTER>**

Esimerkki käynnistää levykkeen formatointiohjelman (toimii vain MSX-DISK-BASIC:ssä).

CLEAR

MUOTO: **CLEAR [määrä] [,end addr]**

VERSIOT: **MSX: KASET. + LEVYKE**
SVI: KASET. + LEVYKE

TARKOITUS: Käskyllä varataan tilaa muuttujille ja/tai asetetaan BASIC-ohjelmalle suurin mahdollinen osoite RAM-muistissa ja suljetaan kaikki avoinna olevat tiedostot. Lisäksi kaikki numeeriset- ja merkkijonomuuttujat "nollataan".

LISÄYKSET: <määrä> = varattavan muistitilan suuruus. Varattu tila on muuttujien, taulukoiden ja konekielisten ohjelmien käytössä. Oletusarvona on 200.
<end addr> = suurin BASIC-tulkin käytettävissä oleva osoite.
Mikäli ohjelmassa käytetään useita tai pitkiä merkkijonoja, on niille varattava ohjelman alussa tilaa CLEAR-käskyllä. Jos haluat käyttää merkkijonoja, joiden pituus ylittää 200 merkkiä, on jokaista käytettävää merkkijonoa kohti varattava haluttu määrä RAM-muistia (merkkijonon maksimi pituus on 254 merkkiä).
Jos halutaan estää ohjelmassa käytettävien muuttujien ja vakioiden mahdollinen päällekirjoitus (esim. tekstinkäsittely- ja taulukkolaskentaohjelmat), on ohjelman alussa määriteltävä suurin käytettävissä oleva osoite.

ESIM:

```
10  CLEAR 508
20  PRINT "TÄMÄ OHJELMA SALLII SINUN"
30  PRINT "TALLENTAA KAKSI TÄYSIPIT"
40  PRINT "KÄÄ MERKKIJONOA MUISTIIN."
```

```
50  PRINT
60  INPUT "ANNA MERKKIJONO 1 "; A$
70  PRINT
80  INPUT "ANNA MERKKIJONO 2 "; B$
90  PRINT
100 PRINT A$
110 PRINT B$
120 END
```

KÄSKY	TOIMINTA
CLEAR	; nollaa kaikki muuttujat ja vakiot, mutta ei poista ohjelmaa muistista.
CLEAR 20000	; nollaa kaikki muuttujat ja vakiot ja varaa BASIC-ohjelmalle tilaa 20 ktavua.
CLEAR 15000,13000	; varaa muuttujille tilaa 13 ktavua ja BASIC-ohjelmalle 15 ktavua.

CLICK

MUOTO: CLICK ON/OFF

VERSIOT: MSX: ei käytössä (korvattu SCREEN-käskyllä)
SVI: KASET. + LEVYKE

TARKOITUS: Käskyllä poistetaan (CLICK OFF) näppäintä painettaessa kuuluva merkkiääni tai kytketään merkkiääni päälle (CLICK ON).

LISÄYKSET: Oletusarvona on CLICK ON. Käsky on varmasti hyödyllinen SV-318:n käyttäjille, sillä painettaessa ko. koneen kovakuminäppäimiä tulee olla huolellinen, mutta tämä merkkiääni auttaa tunnistamaan näppäimen painalluksen.

ESIM:

```
10 CLICK ON
20 PRINT "NÄPPÄILE JOKIN MERKKIJONO"
30 PRINT " JA PAINA ENTER."
40 INPUT A$
50 CLICK OFF
60 PRINT "NÄPPÄILE UUDESTAAN JA"
70 PRINT " JA HUOMAA ERO !"
80 INPUT A$
90 END
```

Muista kääntää ennen ohjelman ajoa televisiostasi/monitoristasi ääni kuuluville.

MSX: Vastaavat komennot MSX-BASIC:ssä ovat:

SVI BASIC	MSX BASIC
CLICK ON	SCREEN ,,1
CLICK OFF	SCREEN ,,0

CONT

MUOTO: CONT

VERSIOT: MSX: KASET. + LEVYKE
SVI: KASET. + LEVYKE

TARKOITUS: Käskyn avulla voidaan jatkaa ohjelmien suoritusta Break-tilanteen jälkeen.

LISÄYKSET: Jos ohjelma keskeytetään painamalla CTRL+STOP -painikkeita tai ohjelmassa on kohdattu STOP-käsky, voidaan suoritusta jatkaa oikeasta kohdasta, tietämättä millä rivillä ohjelman suoritus on ollut.
CONT-käskyä käytetään yleensä ohjelman testausvaiheessa ja etsittäessä ohjelmasta virheitä. Jos ohjelma ei toimi toivotulla tavalla, on syytä sijoittaa ohjelmaan STOP-käskyjä ja kun ohjelma-ajo keskeytyy, voidaan PRINT-käskyllä tutkia muuttujien arvoja. Virheen löydyttyä poistetaan STOP-käskyt. Suoritusta voidaan jatkaa myös GOTO-käskyllä, mutta tällöin tulee olla tiedossa rivinumero, jolle palataan.
Käskyä EI voida käyttää, jos ohjelmaa muutetaan ennen CONT-käskyn antamista tai jos ohjelma on päättynyt eli kaikki rivit on suoritettu. Tällöin tulostetaan virheilmoitus "Can't CONTINUE".

ESIM:

```
10   REM LASKETAAN LUKUJA YHTEEN
20   REM JA KATSOTAAN VÄLILLÄ SUMMAA
30   LET SUMMA%=0 : A%=1 : B%=SUMMA%+A%
40   LET A%=SUMMA%+A%
50   LET SUMMA%=A%+B%
60   STOP
70   LET A%=SUMMA%+B%
80   LET SUMMA%=A%+B%
90   PRINT "JA LOPPUSUMMA ON ";SUMMA%
100  END
```

```
RUN <ENTER>
Break in 60
Ok
PRINT SUMMA% <ENTER>
 2
Ok
CONT <ENTER>
 4
Ok
```

Rivit 10 ja 20 ovat ohjelmaa selventäviä huomautuksia, eivätkä vaikuta ohjelman suoritukseen. Rivillä 30 annetaan joukolle muuttujia lähtöarvot. Rivillä 40 annetaan kokonaislukumuuttujalle A arvo, joka määräytyy muuttujien SUMMA ja A summasta (0+1=1). Rivillä 50 saa muuttuja SUMMA uuden arvon (1+1=2). Rivillä 60 pysäytetään ohjelman kulku STOP-käskyllä ja suorituksen ollessa komentotilassa tarkastelemme muuttujan SUMMA sen hetkistä arvoa (2). Ohjelman suoritusta jatketaan CONT-käskyllä. Lopuksi ohjelma tulostaa rivillä 90 olevan PRINT-lauseen mukaisesti muuttujan SUMMA arvon.

DATA

MUOTO: **DATA [muuttuja] [, muuttuja, . . .]**

VERSIOT:
MSX: KASET. + LEVYKE
SVI KASET. + LEVYKE

TARKOITUS: Käskyllä talletetaan ohjelmaan muuttujia ja vakioita, joita voidaan lukea READ-käskyllä.

LISÄYKSET: DATA-lauseet ovat vapaasti sijoiteltavissa, vaikkakin yleensä ne sijoitetaan ohjelman loppuun tai alkuun. DATA-käskyjä ei suoriteta, eivätkä ne vaikuta ohjelman kulkuun hidastavasti tai näkyvästi.
Tiedot talletetaan DATA-lauseisiin erotettuina pilkuilla (,) ja niitä voidaan sijoittaa yhdelle riville rajattomasti (HUOM! rivin pituus EI voi ylittää 255 merkkiä).
DATA-lauseisiin talletetut muuttujat ja vakiot voivat olla numeerisia, aakkosnumeerisia ja merkkijonoja. Mikäli merkkijonot sisältävät pilkkuja, kaksoispisteitä tai välilyöntejä, joilla on merkitystä, tulee merkkijonot erotella lainausmerkeillä ("muuttuja").
DATA-lauseita suositellaan käytettäväksi, mikäli ohjelma sisältää suuren määrän vakiotietoa. Myös jälkeenpäin tehtävät muuttujien muutokset on helpompi suorittaa DATA-lauseisiin, kuin yksittäisiin ohjelmassa siellä täällä sijaitseviin muuttujiin.
DATA-lauseita voidaan lukea vain järjestyksessä, tosin RESTORE-käskyllä voidaan lukujärjestystä muuttaa. Myöskään DATA-lauseiden määrää ei ole rajoitettu.
READ-käskyllä luettavien muuttujien tulee olla samaa muotoa DATA-lauseesta luettavan muuttujan kanssa. Ei siis voida lukea kokonaislukumuuttujalle A% merkkijonoa "MSX-BASIC".
DATA-lauseet voidaan ajatella tietojonoiksi, joita luetaan READ-käskyllä. Ohjelman ensimmäinen READ-käsky lukee ensimmäisen DATA-lauseen ensimmäisen muuttujan jne. Mikäli kaikki DATA:t on luettu ja READ-käsky silti suoritetaan, annetaan virheilmoitus "Out of DATA". Suositeltavin paikka sijoittaa DATA-lauseet on ohjelman loppu.
HUOM: DATA-lauseet voivat sijaita myös END-käskyn jälkeen.

ESIM:

```
10   REM LUETAAN 5 DATAA
20   FOR A=1 TO 5
30   READ B$(A)
40   NEXT
50   PRINT "OLEMME LUKENEET DATAT JA"
60   PRINT " NYT TULOSTAMME NE."
70   PRINT
80   PRINT "OSALLISTUNEET HENKILÖT OLIVAT:"
90   FOR A=1 TO 5
```

```
100 PRINT B$(A) ;" ";
110 NEXT
120 END
130 DATA VILLE,NIILO
140 DATA KALLE
150 DATA TUIJA,PIRKKO
```

Rivillä 10 on kommentti REM-lauseessa. Riveillä 20—40 on nk. FOR-NEXT -silmukka, joka tällä kertaa lukee viisi (5) merkkijonoa riveiltä 130—150 indeksoiduille muuttujille B$(1)—B$(5). Riveillä 50—80 tulostetaan tekstejä PRINT-käskyillä. Rivit 90—110 muodostavat ohjelmassa toisen FOR-NEXT -silmukan, joka tällä kertaa tulostaa aikaisemmin luetut DATA:t. Rivillä 120 on ohjelman lopetuskäsky END.
HUOM! DATA-lauseet voidaan siis sijoittaa END-käskyn jälkeen.

DEF FN

MUOTO: **DEF FN <nimi> [(<muuttujalista>)] = <funktion määritys>**

VERSIOT: **MSX: KASET. + LEVYKE**
SVI: KASET. + LEVYKE

TARKOITUS: Käskyllä määritellään käyttäjän oma funktio, jossa sallitaan kaikki aritmettiset- ja merkkijonofunktiot.

LISÄYKSET: Funktio on määriteltävä ennen sen käyttöä ja yleensä funktioiden määrittelyt suoritetaankin ohjelman alussa.
<nimi> = käyttäjän määriteltävissä oleva nimi funktiolle. Nimen tulee täyttää normaalille muuttujanimelle asetetut määräykset.
<muuttujalista> = lista muuttujista, joita funktio käyttää. Muuttujat erotetaan toisistaan pilkulla.
<funktion määritys> = lista operaatioista, jotka kohdistuvat ilmaisun <muuttujalista> muuttujiin.
DEF FN -funktiota Ei voi käyttää suorassa toimintamuodossa. Ja mikäli funktio kutsutaan ennen sen määrittelyä, tulostuu virheilmoitus "Undefined user function in XXXX" (XXXX on kyseinen rivinumero).
Käskyä käytetään silloin, kun ohjelmassa suoritetaan jokin useamman tai yhden funktion lause monta kertaa. Näin säästetään tilaa ja aikaa, kun määritelty funktio voidaan kutsua yksinkertaisesti käskyllä FN <nimi> (muuttujalista). Ilmaisujen <muuttujalista>- ja <funktionmääritys>-lauseissa tulee olla samaa tyyppiä. DEF FN -käskyllä voidaan määritellä sekä numeerisia- että merkkijonofunktioita, mutta vain yhtä tyyppiä kerrallaan. Rivi, jolla määrittely tapahtuu, EI saa ylittää 255 merkkiä.

ESIM:

```
10 REM SUORITETAAN FUNKTION MÄÄRITYS
20 DEF FN TILA (SADE,HIGH) = 4*ATN(1)/3*(2*
   SADE-HIGH)pot2*(SADE+HIGH)
30 INPUT "ANNA KAPPALEEN SÄDE "; SADE
40 INPUT "ANNA KORKEUS METREINÄ "; HIGH
50 IF HIGH>2*SADE THEN PRINT "EN VOI LASKEA NÄIN SUURELLA
   KORKEUDELLA,": PRINT "JOTEN OTETAAN UUSIKSI.":GOTO 30
60 PRINT:PRINT
70 PRINT "KAPPALEEN TILAVUUS ON ";
80 PRINT USING "# # # # # #.# # #"; FN KERTO(SADE,HIGH)
90 END
```

Rivillä 10 on selventävä REM-lause (ei vaikuta ohjelman suoritukseen). Rivillä 20 suoritetaan funktion määrittely (lieriömäisen kappaleen tilavuuden laskentakaava). Funktio saa nimeksi 'TILA' (eli FN TILA). Kaavassa käytetään kahta (2) muuttujaa, jotka on nimetty sulkulausekkeessa nimen jälkeen. Riveillä 30 ja 40 kysytään lieriön sädettä ja korkeutta. Rivillä 50 suoritetaan tarkistus, sillä jos korkeus on suurempi kuin 2* annettu säde, saadaan negatiivinen tilavuus, joka ei tietenkään ole mahdollista. Jos annetut mitat ovat sallituissa rajoissa, jatkuu ohjelma riviltä 60, jolla tulostetaan kaksi (2) tyhjää riviä ja jos mitat eivät täyttäneet ehtoa, palataan riville 30. Lopuksi tulostetaan tilavuus käyttäen apuna PRINT USING-lausetta.

```
10 CLS
20 REM ** SINIKÄYRÄ **
30 COLOR 7,1,1
40 SCREEN 2 : REM SVI-BASIC SCREEN 1
50 DEF FN A(X!)=SIN(X!)
60 DEF FN B(X!)=.25
```

```
70  LINE (0,96)—(254,96),7
80  FOR X!=.1 TO 25.5 STEP .1
90  Y!=104—40*(FN A(X!)+FN B(X!))
100 IF Y!<=0 THEN Y!=0
110 IF Y!>=191 THEN Y!=191
120 LINE (X!*10,Y!)—(X!*10,96),15
130 NEXT
140 GOTO 140
```

Tässä esimerkissä piirrämme kuvaruudulle sinikäyrän. Mikäli teet tämän ohjelman SPECTRAVIDEO-tietokoneella, vaihda rivin 40 käsky kuten REM-lauseessa on annettu. Rivillä 50 määritellään käyrän yhtälö. Rivillä 60 oleva määrittely kohdistuu pystysuuntaiseen kertoimeen, jota muuttamalla saat käyrästäsi joko jyrkemmän tai loivemman. Rivillä 70 piirretään X-akseli. Rivit 80—130 muodostavat FOR-NECT-silmukan, jonka aikana käyrä piirretään. Riveillä 100 ja 110 suoritetaan Y-koordinaatille tarkistus, jottei kuva mene kuvaruudun ulkopuolelle. Rivillä 120 suoritetaan käyrän jokaisen pylvään (käyrä muodostetaan pylväistä) piirto. Rivillä 140 annetaan koneelle käsky jäädä tulostamaan tehtyä kuviota ja komentotilaan päästään painamalla CTRL+STOP -näppäimiä.

DEFINT

MUOTO: **DEFINT <muuttuja> [,<muuttuja>, . . .]**

VERSIOT: **MSX: KASET. + LEVYKE**
SVI: KASET. + LEVYKE

TARKOITUS: Käskyllä määritellään yksi tai useampi muuttuja kokonaislukumuuttujaksi.

LISÄYKSET: MSX- ja SVI-BASIC olettaa kaikkien numeeristen muuttujien olevan kaksinkertaisentarkkuudenmuuttujia, ellei toisin DEF-käskyllä määritellä. Muuttuja voidaan määritellä myös sitä käytettäessä joka kerta erikseen muuttujamäärittelyä (ja vakio-) varten varatulla merkillä (% = kokonaisluku, ! = yksinkertaisentark. , # = kaksinkertaisentarkkuuden ja $ = merkkijonomuuttuja/vakio).
Ilmaisulla <muuttuja> määritellään muuttuja kokonaisluvuksi, joten käytettäessä ko. muuttujaa ei tarvita %-merkkiä.
Ilmaisulla <muuttuja> annettu muuttujanimi koskee kaikkia muuttujia, jotka alkavat samalla kirjainyhdistelmällä!! Joten määrittely DEFINT A esimerkiksi koskee seuraavia muuttujia: A, AA, AD, A1, ALKU, jne.

ESIM:

KÄSKY	**MÄÄRITTELEE**
DEFINT A	**; kaikki A-kirjain alkuiset muuttujat kokonaislukumuuttujiksi.**
DEFINT A, F—J	**; kaikki kirjaimilla A,F,G,H,I ja J alkavat muuttujat kok.luk.m.**

Kokonaisluvuksi muunnettaessa suoritetaan tarvittaessa pyöristys.

DEFSNG

MUOTO: **DEFSNG <muuttuja> [, <muuttuja>, . . .]**

VERSIOT: **MSX: KASET. + LEVYKE**
SVI: KASET. + LEVYKE

TARKOITUS: Käskyllä määritellään muuttuja yksinkertaisen tarkkuuden muuttujaksi.

LISÄYKSET: DEFSNG-käskylle pätee samat määrittelyt, kuin DEFINT-käskylle (vrt. DEFINT).

ESIM:

KÄSKY	**MÄÄRITTELEE**
DEFSNG A	**; kaikki A-alkuiset muuttujat yksinkertaisen tarkkuuden muuttujiksi.**
DEFSNG D—G	**; kaikki D,E,F ja G kirjaimella alkavat muuttujat yk. tark. m.**

Yksinkertaisen tarkkuuden muuttuja tulostetaan käyttäen seitsemää merkitsevää numeroa. Tarvittaessa suoritetaan viimeiselle numerolle pyöristys ylöspäin.

DEFDBL

MUOTO: **DEFDBL <muuttuja> [, <muuttuja>, . . .]**

VERSIOT: **MSX: KASET. + LEVYKE**
SVI: KASET. + LEVYKE

TARKOITUS: Käskyllä määritellään muuttuja kaksinkertaisen tarkkuuden muuttujaksi.

LISÄYKSET: Käsky voidaan korvata #-merkillä. DEFDBL-käskylle pätee samat määritykset, kuin DEFINT-käskylle (vrt. DEFINT).

ESIM:

KÄSKY	MÄÄRITTELEE
DEFDBL A	**; kaikki kirjaimella A alkavat muuttujat kaksinkertaisen tarkkuuden muuttujiksi.**
DEFDBL R—T	**; kaikki R,S ja T kirjaimella alkavat muuttujat kak. tark. m.**

DEFSTR

MUOTO: **DEFSTR <muuttuja> [,<muuttuja>, . . .]**

VERSIOT: **MSX: KASET. + LEVYKE**
SVI: KASET. + LEVYKE

TARKOITUS: Käskyllä määritellään annetut muuttujat merkkijonomuuttujiksi.

LISÄYKSET: Käytettäessä ko. käskyä voidaan sillä määritellyistä merkkijonomuuttujista jättää $-merkki pois. DEFSTR-käskyyn pätee samat määrittelyt, kuin DEFINT-käskyyn.

ESIM:

KÄSKY	MÄÄRITTELEE
DEFSTR A	**; kaikki kirjaimella A alkavat muuttujat merkkijonomuuttujiksi.**
DEFSTR F—H	**; kaikki kirjaimilla F,G ja H alkavat muuttujat merk.jonom.**

DEFUSR

MUOTO: **DEFUSR [<num>] = <osoite>**

VERSIOT: **MSX: KASET. + LEVYKE**
SVI: KASET. + LEVYKE

TARKOITUS: Käskyllä määritellään konekielisen aliohjelman osoite.

LISÄYKSET: <num> = konekielisen aliohjelman numero. Kokonaisluku väliltä 0—9.
<osoite> = osoite, josta konekielinen aliohjelma alkaa. Voidaan antaa heksa-, oktaali- tai desimaalilukuna.
Ilmaisu <num> voidaan jättää antamatta, jolloin aliohjelman numeroksi oletetaan nolla (0).
Käytettäessä BASIC-ohjelmien apuna konekielisiä aliohjelmia, on ennen aliohjelmiin siirtymistä määriteltävä kunkin aliohjelman alkuosoite. Konekielisten aliohjelmien määrää ei ole rajoitettu, mutta kerrallaan voi olla määriteltynä 10 kpl.
Otamme esimerkiksi ROM-muistissa olevan aliohjelman, joka vastaa CLS-käskyä.

ESIM:

```
10  DEFUSR = &HC3
20  A=USR(0)
```

Kokeile ohjelmaa kirjoittamalla se muistiin ja anna RUN-käsky. Vaikutus on sama, kuin CLS-käskyllä. Ohjelma toimii vain MSX-BASIC:ssä.
MSX- ja SVI-BASIC ei tarvitse välttämättä konekielistä ohjelmointia, sillä kieli sisältää grafiikka- ym. käskyt. Mikäli halutaan nopeuttaa rutiineja, löytyy konekielestä apua.

DELETE

MUOTO: **DELETE [<num1>] [— <num2>]**

VERSIOT: **MSX: KASET. + LEVYKE**
SVI: KASET. + LEVYKE

TARKOITUS: Käskyllä poistetaan muistista ohjelmarivejä.

LISÄYKSET: <num1> = rivinumero, josta poistaminen aloitetaan. Voidaan jättää antamatta.
<num2> = rivinumero, johon asti poisto tapahtuu. Voidaan korvata pisteellä (.).
Käskyllä voidaan poistaa rivi, rivit johonkin rivinumeroon asti, joltakin väliltä tai koko ohjelma. Yhden tietyn rivin poisto käy nopeasti kirjoittamalla kyseinen rivinumero ja painamalla ENTER-painiketta.
Käsky 'DELETE.' poistaa viimeksi käsitellyn rivin eli rivin, jota viimeksi on muokattu, listattu, joka on viimeksi syötetty tai jolla on tapahtunut virhe.

ESIM:

KÄSKY	**POISTAA RIVIT**
DELETE 120	**; rivin 120.**
DELETE —120	**; rivit 0—120.**
DELETE 10—120	**; rivit 10—120.**
DELETE.	**; rivin, jota on käsitelty viimeksi.**
120 <ENTER>	**; rivin 120 (HUOM! rivi 120 tulee ennen käskyn antamista olla muistissa).**
DELETE —.	**; poistaa kaikki rivit (riviltä nolla alkaen) viimeksi käsitellylle riville asti.**
DELETE.—120	**; poistaa alkaen viimeksi käsitellyltä riviltä riville 120 asti kaikki rivit (mukaanlukien em. rivit).**

DIM

MUOTO: **DIM <muuttuja>(<määrä>) [,<muuttuja2>(<määrä2>)]**

VERSIOT: **MSX: KASET. + LEVYKE**
SVI: KASET. + LEVYKE

TARKOITUS: Käskyllä määritellään numeerisen- tai merkkijonomuuttujan tai taulukon eli matriisin maksimi koko.

LISÄYKSET: <muuttuja> = muuttujanimi, jolle tilaa varataan.
<määrä> = varattavan muistitilan määrä. Oletusarvo 10.
Jos ohjelman alussa tai viimeistään ennen taulukon/matriisin käyttöä ei DIM-käskyllä taulukkoa määritellä, olettaa kone taulukon indekseiksi (syyvyydeksi) joka suuntaan 10. Koska taulukon pienin mahdollinen alkio on nolla (0), on käytössä indeksit 0—10 eli 11 kpl. Mikäli ei haluta varata näin paljoa tilaa tai tämä tila ei riitä, määritellään DIM-lauseella taulukolle uusi koko.
DIM-käskyn jälkeen tulee kaikkien taulukon alkioiden alkuarvoiksi nolla (merkkijonotaulukoilla tyhjä). Käytettäessä useampi ulotteisia taulukoita kuin yksi ulotteinen, merkitään ilmaisussa määrä jokainen dimensio eli syvyys erikseen pilkuilla erotettuina. Jos käytettäisiin esim. kaksiulotteista taulukkoa, käsky DIM A(4,12) varaisi alkioille (0,1,2,3,4) ja (0,1,. . .,11,12) tilaa taulukosta, jonka nimi on A ja joka sisältää numeerisia arvoja.

ESIM:

KÄSKY	**LUO MATRIISIN**
DIM A(7)	**; jonka koko on kahdeksan (8) alkiota ja on tarkoitettu numeerisille muuttujille.**
DIM B$(20,50)	**; jonka koko on 21*51 alkiota ja joka on tarkoitettu merkkijonojen tallentamiseen.**
DIM C%(3,3,5)	**; jonka koko on 4*4*6 alkiota ja joka on tarkoitettu kokonaislukujen tallennukseen. HUOM! kolmiulotteinen matriisi.**

Yhdellä DIM-käskyllä voi varata saman aikaisesti useammalle taulukolle/matriisille tilaa.

ESIM: **DIM A%(6), B!(5), C#(3), FF$(12,5)**

varaisi tilaa seuraaville matriiseille:
1. matriisille, jonka koko on seitsemän (7) alkiota ja joka on tarkoitettu kokonaisluvuille.
2. matriisille, jonka koko on kuusi (6) alkiota (nimi B) ja joka olisi yksinkertaisen tarkkuuden luvuille.
3. matriisille nimeltään C, jonka koko on neljä (4) alkiota ja joka on tarkoitettu kaksinkertaisentarkkuuden luvuille.
4. kaksiulotteiselle merkkijonomatriisille (nimi FF), jonka alkioiden määrä on 13*6 kpl.

Matriisin koko ei tarvitse olla tiedossa etukäteen, vaan tiedot matriisin koosta voidaan syöttää muuttujien avulla.

ESIM:

```
200 INPUT "ANNA MATRIISIN KOKO "; A,B,C
210 DIM A$(A,B,C)
```

Tässä esimerkissä pyydetään rivillä 200 INPUT-käskyn avulla kolmea matriisin indeksiä, jotka rivillä 210 DIM-käskyssä syötetään.

END

MUOTO: END

VERSIOT: MSX: KASET. + LEVYKE
SVI: KASET. + LEVYKE

TARKOITUS: Käskyllä päätetään ohjelma ja palataan komentotilaan.

LISÄYKSET: Käsky sulkee kaikki avoinna olevat tiedostot.
Käsky ei ole pakollinen, sillä BASIC olettaa rivillä 65529 olevan aina END-käskyn. Toisin kuin CTRL+STOP näppäinten painaminen tai STOP-käsky, ei END-käskyn jälkeen tulosteta "Break in XXXX" -tekstiä.
END-käskyn käyttö ei siis ole pakollista, mutta toisinaan sitä on syytä käyttää, ettei ajauduta riville, joka aiheuttaisi virheen.

ESIM:

```
10  PRINT "LASKETAAN IKÄ PÄIVISSÄ"
20  PRINT "KÄYTTÄEN APUNA ALIOH-"
30  PRINT " JELMA KÄSKYÄ GOSUB."
40  PRINT
50  INPUT "ANNA IKÄSI VUOSISSA ";A
60  GOSUB 100
70  PRINT
80  PRINT "IKÄSI PÄIVISSÄ ON";A;"VUOTTA."
90  END
100 LET A=A*365
110 RETURN
```

Rivit 10—40 tulostavat ohjelman alussa ohjeita. Rivillä 50 pyytää kone syöttämään ikää INPUT-käskyllä. Rivillä 60 hypätään aliohjelmaan riville 100, jolla muuttuja A eli ikä saa uuden arvon, joka on ikä päivissä (ikä vuosina * 365 päivää). Rivillä 110 annetaan käsky palata pääohjelmaan ja se tapahtuu riville 70. Rivit 70—80 antavat laskennan tuloksen. Rivillä 90 pysäytetään ohjelma, jotta ei vahingossa kohdattaisi rivillä 110 olevaa RETURN-käskyä. Tämä aiheuttaisi virheen (kts. GOSUB-käsky).

ERASE

MUOTO: ERASE <muuttuja> [,<muuttuja>, . . .]

VERSIOT: MSX: KASET. + LEVYKE
SVI: KASET. + LEVYKE

TARKOITUS: Käskyllä poistetaan tarpeettomia taulukoita/matriiseja ja vapautetaan muuttujanimet.

LISÄYKSET: <muuttuja> = vapautettavan muuttujan/matriisin nimi.
Käskyä käytetään silloin, kun ohjelmassa käytettyä taulukkoa/matriisia ei enää tarvita ja sen viemä muistitila tarvitaan muuhun käyttöön.
Jos tuhotut taulukot halutaan määritellä uudestaan, käy se normaalisti DIM-käskyllä. Jo tuhottuja taulukoita se ei kuitenkaan tuo takaisin.

ESIM:

```
10  DIM A$(10,15)
        . . .
200 ERASE A$
210 DIM A$(5,12)
220 DIM B$(5,12)
```

Tässä esimerkissä ohjelman alussa on DIM-käsky, jolla määritellään kaksiulotteinen merkkijonomatriisi A. Myöhemmin ohjelmassa tarvitaan rivillä 10 määritellyn taulukon tarvitsema muistitila käyttöön (myös matriisin tietoja ei enää tarvita). Näin saatu muistitila käytetään kahdelle pienemmälle matriisille, joista toinen saa vanhan nimen A.

ERROR

MUOTO: ERROR <num>

VERSIOT: MSX: KASET. + LEVYKE
SVI: KASET. + LEVYKE

TARKOITUS: Käskyä käytetään tuottamaan tietty virhekoodi tai antamaan käyttäjän itse määriteltävissä oleva virhekoodi.

LISÄYKSET: <num> = kokonaisluku väliltä 1—255, jolla määritellään joko oma virhekoodi tai tuotetaan valmiiksi BASIC:n määrittelemä virhekoodi.
Käskyä käytetään virhekoodien testauksessa. Mikäli ilmaisu <num> on väliltä 1—59, on kyseessä BASIC:n määrittelemä virhe ja jos <num> on väliltä 60—255, on käytössä käyttäjän oma virhekoodinumero.
Kohdatessaan ERROR-käskyn laskee kone ilmaisun <num> arvon (ilmaisu <num> voi siis olla myös matemaattinen lause, arvona kuitenkin kokonaisluku). Jos saatu arvo on välillä 1—255, on toiminto sama kuin rivillä olisi tapahtunut virhe.
Käskyä käytetään ON ERROR GOTO -lauseen kanssa, jotta virheenkäsittely tapahtuisi hallitusti. Mikäli ilmaisun <num> arvo EI ole välillä 1—255, on kyseessä käyttäjän virhe ja suoritus keskeytyy tai jää mahdollisesti pyörimään ikuiseen silmukkaan, joten on syytä olla huolellinen.
HUOM! MSX-BASIC ja SVI-BASIC eroaa virhekoodeista 58—60!

ESIM:

```
10   ON ERROR GOTO 1000
20   REM ** OHJELMA ALKAA TÄSTÄ **
              . . .
60   PRINT "VASTAA SEURAAVAAN KYSYMYKSEEN"
70   PRINT "Y, JOS HALUAT NÄHDÄ VIRHERUTIIN"
80   PRINT "        NIN TOIMIVAN."
90   INPUT "JA VASTAUKSESI";ANS$
100  IF ANS$="Y" OR ANS$="y" THEN ERROR 77
110  END
              . . .
1000 PRINT "TOIMII !":RESUME 110
```

Rivillä 10 määritellään rivinumero (1000), jolle siirrytään virheen sattuessa. Rivillä 20 on selventävä huomautus REM-lauseessa. Riveille 21—59 voidaan sijoittaa muuta ohjelmaa tai ne voidaan unohtaa (kuten tässä). Rivit 60—80 sisältävät ohjeita PRINT-lauseissa. Rivillä 90 pyydetään käyttäjää vastaamaan joko "Y" (tai "y") tai jotain muuta, jolloin ohjelma päättyy. Jos vastataan myönteisesti, tuotetaan virhe rivillä 110 (virhekoodiksi merkitään 77, joka on vapaasti käytettävissä oleva virhekoodinumero). Tällöin (koska ohjelman alussa määriteltiin), siirtyy suoritus riville 1000. Rivillä 1000 tulostetaan ensin teksti "TOIMII!" ja sitten palataan riville 110 oikealla käskyllä.
ERROR-käskyä voit käyttää myös komentotilassa tulostamaan virhekoodeja.

ESIM:

```
ERROR 2 <ENTER>
Syntax error
Ok
```

FOR . . . NEXT

MUOTO: **FOR <muuttuja>=<luku1> TO <luku2> [STEP <luku3>]**
NEXT <muuttuja> [,<muuttuja>, . . .]

VERSIOT: **MSX: KASET. + LEVYKE**
SVI: KASET. + LEVYKE

TARKOITUS: FOR-NEXT -silmukalla toistetaan jokin käsky tai osa ohjelmaa haluttu määrä kertoja.

LISÄYKSET: <muuttuja> = toistorakenteen laskuri. Tätä muuttujaa kasvatetaan joka kierroksella ilmaisun <luku3> määräämällä arvolla.
<luku1> = arvo, jonka kierroslaskuri (<muuttuja>) ensimmäisellä kerralla saa.
<luku2> = arvo, jonka kierroslaskuri saa viimeisellä kierroksella.
<luku3> = arvo, jolla kierroslaskuria joka kierroksella kasvatetaan tai vähennetään.
Silmukan ensimmäisellä kierroksella saa ilmaisu <muuttuja> arvon <luku1> ja kaikki käskyt NEXT-käskyyn asti suoritetaan. Tällöin laskurin (<muuttuja>) arvoon lisätään ilmaisun <luku3> arvo (jos <luku3> on negatiivinen luku, suoritetaan vähennys) tai <luku3>:n puuttuessa yksi (oletusarvo).
Askel eli ilmaisu <luku3> voi siis olla joko positiivinen tai negatiivinen luku. Jos <luku3>on annettu ja se on positiivinen, tulee ilmaisun <luku2> olla suurempi kuin ilmaisun <luku1> ja päinvastoin.
Kun silmukassa kohdataan NEXT-käsky, kasvatetaan ensin laskuria <muuttuja> ja sitten suoritetaan tarkistus onko laskurin arvo suurempi (positiivinen silmukka) ja jos on, poistutaan silmukasta ja suoritetaan NEXT-käskyä seuraava ohjelmarivi.
FOR-NEXT -silmukoita voi olla myös sisäkkäin. Niiden määrällä ei ole rajoitusta, mutta mikäli silmukoita on käytössä useita, tulee olla huolellinen, sillä silmukat on suljettava oikeassa järjestyksessä. NEXT-käskyssä ei tarvitse välttämättä ilmaista suljettavan silmukan muuttujaa, sillä BASIC osaa tehdä sen itse. Tällöin sulkeutuu viimeksi avattu silmukka.

Jos samalla rivillä suljetaan useita silmukoita, voidaan kaikki silmukat sulkea yhdellä NEXT-käskyllä. Tällöin annetaan suljettavien silmukoiden muuttujat (laskurit eli ilmaisut <muuttuja>) erotettuina pilkuilla tai jätetään antamatta (taataan oikea sulkemisjärjestys). Jos silmukoita on sisäkkäin useampia, sulkeutuu aina ensimmäiseksi avattu silmukka viimeisenä.

ESIM:

```
10   FOR A=1 TO 3
20   PRINT "SILMUKKALASKURIN ARVO ON NYT";A
30   NEXT A
40   PRINT "LASKURIN ARVOKSI JÄÄ";A
50   END
```

```
RUN <ENTER>
SILMUKKALASKURIN ARVO ON NYT 1
SILMUKKALASKURIN ARVO ON NYT 2
SILMUKKALASKURIN ARVO ON NYT 3
LASKURIN ARVOKSI JÄÄ 4
Ok
```

Rivillä 10 määrätään, että laskuri A saa arvot 1—3 ja askel on yksi (ilmaisua <luku3> ei ole määritelty). Ensimmäisellä kierroksella saa laskuri A arvon yksi (1), joka rivillä 20 tulostetaan. Rivillä 30 oleva NEXT-käsky sulkee silmukan ja laskurin A arvoa kasvatetaan yhdellä. Nyt testataan, onko laskurin A arvo suurempi kuin ilmaisun <luku2> ja koska näin ei ensimmäisellä kierroksella ole, palataan FOR-käskystä seuraavalle riville (rivi 20). Näin käy seuraavallakin kierroksella. Kolmannella kierroksella laskuria A on kasvatettu arvoon 4 ja tapahtuu testaus, onko <luku2> < laskurin A arvo. Ja koska näin ei ole, siirrytään riville 40 (HUOM! A:n arvoksi jäi neljä). Rivillä 40 tulostetaan A:n lopullinen arvo (4) ja rivillä 50 päätetään ohjelma.

```
10   FOR A=3 TO 1 STEP —1
20   PRINT "LASKURIN ARVO ON";A
30   NEXT
40   PRINT "LOPULLINEN ARVO ON";A
```

```
RUN <ENTER>
LASKURIN ARVO ON 3
LASKURIN ARVO ON 2
LASKURIN ARVO ON 1
LOPULLINEN ARVO ON 0
Ok
```

Tällä kertaa silmukan laskuria vähennettiin (STEP —1) yhdellä/kierros. Erona myös se, että NEXT-käskyssä ei annettu laskurin muuttujanimeä.

```
10   FOR A=1 TO 1.6 STEP .2 : PRINT "LASKURIN ARVO ON";A : NEXT
20   PRINT "LOPULLINEN ARVO ON";A
30   END
```

```
RUN <ENTER>
LASKURIN ARVO ON 1.6
LASKURIN ARVO ON 1.4
LASKURIN ARVO ON 1.2
LOPULLINEN ARVO ON 1
Ok
```

HUOM! NEXT-käsky ei tarvitse omaa riviä.

```
10   FOR A=1 TO 3
20   FOR B=500 TO 550 STEP 50
30   PRINT "ENSIMMÄISEN LASKURIN ARVO ON";A
40   PRINT "TOISEN LASKURIN ARVO ON";B
50   PRINT
60   NEXT
70   PRINT "LASKURIN YKSI LOPPUARVO ON";A
80   PRINT "LASKURIN KAKSI LOPPUARVO ON";B
90   END
```

Tässä esimerkissä on kaksi silmukkaa, joista toinen on toisen sisällä. Kirjoita ohjelma ja aja se RUN-käskyllä. Huomaa, kuinka rivillä 60 suljetaan molemmat silmukat yhdellä NEXT-käskyllä.

FOR-NEXT -silmukalla voidaan toteuttaa ajallinen viive, joka ei tee mitään muuta kuin kuluttaa aikaa. Esim:

```
10   PRINT "VOIT MÄÄRÄTÄ SILMUKAN KESTON"
20   PRINT "ANTAMALLA JONKIN LUVUN 1—10000."
```

```
30   PRINT
40   INPUT "ANNA KESTOAIKA";LOOP%
50   FOR A%=1 TO LOOP%
60   NEXT
70   PRINT "VALMIS!"
80   END
```

Rivillä 40 pyydetään syöttämään kokonaislukua, joka määrää riveillä 50—60 vietettävän ajan. Huomaa, että jättämällä NEXT-käskystä muuttujan (tällä kertaa LOOP%) pois, saadaan silmukka kiertämään nopeammin. Voit testata sen antamalla ensin suuren luvun (esim. 10000) ja ottamalla silmukasta ajan (käynnistä kello, kun olet kirjoittanut luvun 10000 ja painat ENTER!). Vaihda nyt rivi 60 seuraavaksi:

```
60   NEXT A%
```

Kokeile syöttämällä sama luku (esim. 10000) ja ota aika.

Seuraavaksi esimerkki, jossa osoitetaan FOR-lauseessa käytettävien muuttujien arvojen säilymistä silmukan aikana, vaikka muuttujille yritettäisiin antaa jokin arvo.

```
10   B=1 : C=7 : D=2
20   FOR A=B TO C STEP D
30   B=3 : C=9 : D=5
40   PRINT "LASKURIN ARVO ON";A
50   NEXT
60   END
```

Rivillä 10 annetaan silmukassa käytettäville muuttujille arvot. Rivillä 20 määrätään laskurin nimeksi A ja määritellään silmukan ilmaisut. Rivillä 30 yritetään kesken silmukkaa syöttää muuttujille arvoja, joita ne eivät koskaan saa. Rivillä 40 tulostetaan silmukkalaskurin arvo ja rivin 50 NEXT-käsky sulkee silmukan. Aja ohjelma RUN-käskyllä ja totea sen toimivan moitteettomasti.

Seuraavaksi esimerkki sisäisistä silmukoista.

```
10   FOR A=1 TO 3
20   FOR B=2 TO 4
30   PRINT A;B
40   NEXT : REM * SILMUKKA B *
50   PRINT
60   NEXT : REM * SILMUKKA A *
70   END
```

Huomaa silmukoiden sulkujärjestys!

Silmukoista voidaan poistua väliaikaisesti esim. GOSUB-käskyllä, josta esimerkki.

```
10   FOR D=1 TO 3
20   GOSUB 50
30   NEXT
40   END
50   PRINT "OLEMME RIVILLÄ 50 JA NYT PALAAMME ALIOHJELMASTA."
60   RETURN
```

Huomaa rivillä 40 oleva END-käsky. Mikäli se jätettäisiin pois, saataisiin viimeisen silmukan jälkeen virheilmoitus RETURN-käskystä.

Jos ennalta ei tiedetä, tuleeko STEP:n arvoksi positiivinen vai negatiivinen luku, voidaan automaattinen silmukka toteuttaa seuraavasti.

```
10   FOR LOOP=A TO B STEP 1+2*(A>)
                . . .
100  NEXT
```

Jos A<=B, on lause A>B = EPÄTOSI eli nolla (0) ja näin askeleeksi tulee yksi (1) eli laskenta ylöspäin.
Jos A>B, on lause A>B = TOSI eli —1 ja silloin askeleeksi tulee —1 eli laskenta alaspäin.
Mikäli NEXT-käsky suoritetaan ennen tai ilman FOR-lausetta, tulostuu virheilmoitus "NEXT without FOR".

GOSUB . . . RETURN

MUOTO: GOSUB <nro> . . . RETURN [<num>]

VERSIOT: MSX: KASET. + LEVYKE
SVI: KASET. + LEVYKE

TARKOITUS: GOSUB-käskyllä siirrytään BASIC-kieliseen aliohjelmaan ja RETURN-käskyllä suoritetaan aliohjelmasta paluu.

LISÄYKSET: <nro> = rivinumero, jolle GOSUB-käskyn jälkeen siirrytään.
<num> = rivinumero, jolle paluu aliohjelmasta suoritetaan.
GOSUB-käskyllä suoritetaan hyppy BASIC-kieliseen aliohjelmaan, joka alkaa riviltä <nro>. Aliohjelmien määrää ei ole rajoitettu, vaikkakin teoreettinen raja on olemassa (riippuu monesta tekijästä). Aliohjelmat voivat kutsua toisia aliohjelmia eli puhutaan sisäkkäisistä aliohjelmista. Yhdessä aliohjelmassa voi olla myös useita RETURN-käskyjä, jos tarve vaatii. Aliohjelmat on kuitenkin suojattava niin, ettei niihin päästä muuten kuin GOSUB-käskyllä, sillä muutoin aliohjelman lopussa oleva RETURN-käsky aiheuttaa virheen. Usein aliohjelmat suojataan STOP-, End- tai GOTO-käskyillä.
Aliohjelmia käytetään, jos joukko käskyjä tai osa ohjelmaa suoritetaan useita kertoja. Näin vältytään turhalta toistolta. Yleensä aliohjelmien tehtävänä on hoitaa tiedon keruu, käsittely ja tulostus. Pääohjelman tehtävä on yhdistellä tehokkaita aliohjelmia. Aliohjelmien muuntaminen on selkeämpää, kuin sekavan sinne tänne hyppivän pääohjelman.

ESIM:

```
10  CLS
20  REM ** PÄÄOHJELMA **
30  GOSUB 120
40  REM * KYSYTÄÄN KATEETIT *
50  GOSUB 200
60  REM * LASKETAAN HYPOTENUUSA *
70  GOSUB 240
80  REM * TULOSTETAAN VASTAUS *
90  GOSUB 270
100 REM ** LOPPU **
110 END
120 REM * ALIOHJELMA 1 *
130 PRINT "LASKETAAN SUORAKULMAISEN KOLMION"
140 PRINT "HYPOTENUUSA KÄYTTÄEN LASKEMISESSA"
150 PRINT "ALIOHJELMAA. ENSIN TARVITAAN KA-
160 PRINT "TEETTIEN SIVUJEN PITUUDET."
170 PRINT
180 REM * PALUU *
190 RETURN
200 PRINT "ANNA KATEETIT";A,B
210 PRINT
220 REM * PALUU *
230 RETURN
240 C=SQR(A*A+B*B)
250 REM * PALUU *
260 RETURN
270 PRINT "ANNETUN KOLMION HYPOTENUUSA ON";C
280 REM * PALUU *
290 RETURN 100
```

GOTO

MUOTO: GOTO <nro>

VERSIOT: MSX: KASET. + LEVYKE
SVI: KASET. + LEVYKE

TARKOITUS: Käskyllä määrätään ehdoton hyppy riville numero <nro>.

LISÄYKSET: <nro> = rivinumero, jolle hyppy suoritetaan. Voi olla myös matemaattinen lause, joka ensin suoritetaan ja hyppy suoritetaan lauseesta saadulle kokonaisluvulle (=rivinumero). Mikäli ohjelmassa ei ole rivinumeroa <nro>, tulostetaan virheilmoitus "Undefined line number". GOTO-käskyä voidaan käyttää myös ohjelman käynnistämiseen vaihtoehtona RUN-käskylle. GOTO-käskyn erona on kuitenkin se, että muuttujia EI nollata.
GOTO-käskyä tulisi välttää, koska sillä saadaan aikaan sekaviakin ohjelmarakenteita.

ESIM:

```
10  GOTO 40
20  PRINT "ANTAMASI LUKU KOROTETTUNA TOISEEN ON";A*A
30  END
40  CLS
50  INPUT "ANNA JOKIN LUKU";A
60  PRINT
70  GOTO 20
```

Rivillä 10 hypätään heti riville 40, josta ohjelma jatkuu ruuduntyhjennyskäskyllä (CLS). Seuraavaksi rivillä 50 pyydetään jotain lukua INPUT-käskyllä. Rivillä 60 tulostetaan tyhjä rivi ja rivillä 70 suoritetaan taas hyppy, nyt riville 20. Ohjelma päättyy rivin 20 tulostuksen jälkeen riville 30. Melko sekava, vai mitä?

IF . . . THEN . . . ELSE

MUOTO: **IF <lause> THEN <käsky> / <nro> [: <käsky> : . . .]**
[ELSE <order> / <nro>] [: <order> : . . .]

VERSIOT: **MSX: KASET. + LEVYKE**
SVI: KASET. + LEVYKE

TARKOITUS: Käskyllä annetaan ehtolause, jonka testauksen tuloksena suoritetaan käsky/-jä.

LISÄYKSET: <lause> = joko matemaattinen yhtälö tai merkkijonon testaus.
<käsky> = käsky, joka suoritetaan mikäli ilmaisun <lause> ehto toteutuu. Voi olla myös rivinumero (<nro>), jolta ohjelman suoritus jatkuu.
<order> = käsky, joka suoritetaan mikäli ilmaisun <lause> ehto ei toteudu (lause on EPÄTOSI).
Jos ilmaisu <lause> on TOSI, suoritetaan kaikki käskyt (joita voi olla useita kaksoispisteellä erotettuina) ELSE-käskyyn asti. Jos ELSE-komentoa ei anneta, suoritetaan kaikki IF—THEN -lauseessa THEN-sanan jäljessä olevat käskyt. Jos <lause> on EPÄTOSI, suoritetaan ELSE-sanan jäljessä olevat käskyt rivin loppuun asti tai ELSE-sanan puuttuessa siirrytään seuraavalle riville.
IF-lauseita voidaan sijoittaa myös sisäkkäin, jolloin ELSE-lause viittaa lähimpään IF-lauseeseen.

ESIM:

```
10  PRINT "ANNA LUKU 1, JOS OLET TYTTÖ"
20  PRINT "TAI LUKU 2, JOS OLET POIKA."
30  PRINT
40  INPUT A
50  REM ** SUORITETAAN TESTAUS **
60  IF A=1 THEN PRINT "OLET TYTTÖ."
70  IF A=2 THEN PRINT "OLET POIKA."
80  END
```

Riveillä 10 ja 20 tulostetaan käyttäjälle ohjeita ja rivillä 30 yksi tyhjä rivi. Rivillä 40 pyytää kone lukua, jolla merkitään muistiin onko käyttäjä tyttö vai poika. REM-käsky rivillä 50 on selventävänä huomautuksena, eikä vaikuta ohjelman suoritukseen. Riveillä 60 ja 70 tapahtuu muuttujan A testaus. Jos A:lle on syötetty arvo 1, saa rivin 60 IF-lause arvon tosi ja tulostetaan THEN-käskyn jäljessä oleva PRINT-lause. Jos A < > 1, silloin siirrytään riville 70 testaamaan, onko A:n arvo 2. Jos on, tulostuu PRINT-lause, jos ei, siirrytään riville 80, jolla ohjelma päätetään käskyyn END (ei pakollinen). Jos A:n arvoksi syötetään jokin muu luku kuin 1 tai 2, ei rivin 60 eikä rivin 70 ehtolause toteudu ja tässä tapauksessa ei tulosteta mitään.
Rivit 60 ja 70 voidaan yhdistää seuraavasti:

```
60  IF A=1 THEN PRINT "OLET TYTTÖ." ELSE IF A=2 THEN PRINT
    "OLET POIKA."
70  REM * RIVI MITÄTÖIDÄÄN REM-LAUSEELLA *
```

Jos A:lle on annettu arvo 1, toteutuu ensimmäinen ehtolause ja tulostuu "OLET TYTTÖ.", jonka jälkeen siirrytään seuraavalle riville. Jos taas A:n arvoksi on syötetty 2, toteutuu jälkimmäinen ehtolause. Jos A:lle on syötetty jokin muu luku, ei kumpikaan ehtolauseista toteudu. Rivi 70 on poistettava tai eliminoitava REM-käskyllä, jottei suoritettaisi kahta vertailua (kts. edellisen esimerkin rivi 70).
Esimerkki sisäkkäisistä ehtolauseista:

```
100 IF A=1 THEN IF B=2 THEN GOTO 200 ELSE GOTO 300
110 REM  OHJELMA JATKUU TÄLTÄ RIVILTÄ, JOS A < > 1
         . . .
200 REM  OHJELMA JATKUU TÄLTÄ RIVILTÄ, JOS A=1 JA B=2
         . . .
300 REM  OHJELMA JATKUU TÄLTÄ RIVILTÄ, JOS A=1 JA B < > 2
```

IF ... GOTO ... ELSE

MUOTO: **IF <lause> GOTO <nro>**
[ELSE <käsky>/<nro>] [: <käsky> : . . .]

VERSIOT: **MSX: KASET. + LEVYKE**
SVI: KASET. + LEVYKE

TARKOITUS: Käskyllä annetaan ehtolause, jonka testauksen tuloksena siirrytään halutulle riville.

LISÄYKSET: >lause> = testattava lause, joka voi olla matemaattinen tai merkkijonon testaus.
<nro> = rivinumero, jolle suoritus siirtyy, jos IF-lauseen ehto toteutuu.
<käsky> = käsky, joka suoritetaan, jos IF-lauseen testin tulos on EPÄTOSI.
IF—GOTO—ELSE -lause toimii kuten IF—THEN—ELSE -lause, mutta erona on IF—GOTO—ELSE -lauseessa välitön ohjelmariville siirtyminen. Myös IF—GOTO -lauseita voidaan sisäistää useita peräkkäin. Kts. IF—THEN—ELSE -lause.

ESIM:

```
10  PRINT "ANNA LUKU 1, JOS OLET TYTTÖ"
20  PRINT "TAI LUKU 2, JOS OLET POIKA."
30  PRINT
40  INPUT A
50  IF A=1 GOTO 80
60  IF A=2 GOTO 100
70  END
80  PRINT "OLET TYTTÖ."
90  END
100 PRINT "OLET POIKA."
110 END
```

Jos A:n arvoksi syötetään jokin muu luku, kuin 1 tai 2, pysähtyy ohjelma riville 70 END-käskyyn, sillä rivin 50 eikä rivin 60 ehto toteudu. Riveillä 70 ja 90 käytetään END-käskyä, jotta vältyttäisiin virheelliseltä PRINT-lauseelta.

INPUT

MUOTO: **INPUT ["<teksti>";] <muuttuja> [,<muuttuja>, . . .]**

VERSIOT: **MSX: KASET. + LEVYKE**
SVI: KASET. + LEVYKE

TARKOITUS: Käsky sallii muuttujien ja/tai merkkijonon syötön ohjelman aikana.

LISÄYKSET: >teksti> = INPUT-lause sallii PRINT-käskyn yhdistämisen antamalla mahdollisuuden liittää kysymysmerkkiä ennen selventävää tekstiä. Ilmaisu <teksti> voi olla mitä tahansa aakkosnumeerista tietoa.
<muuttuja> = muuttujanimi, jolle INPUT-käskyllä tietoa halutaan syöttää. Voi olla merkkijono- tai numeerinen muuttuja. Muuttujia voidaan yhteen INPUT-lauseeseen sijoittaa rajattomasti (ohjelmarivin pituus ei voi ylittää 255 merkkiä!), jolloin muuttujat erotetaan pilkuilla.
INPUT-lause lukee muuttujaan kaikki merkit aina ENTER:n painamiseen asti ja syötettävän muuttujan editointi syötön aikana on mahdollista. Merkkijono, joka kuvaruudulla näkyy ENTER:ä painettaessa, syöttyy muuttujalle.
Kohdatessaan INPUT-lauseen tulostaa BASIC ensin ilmaisussa <teksti> (mikäli sitä on käytetty) olevan tekstiosan ja heti tämän perään kysymysmerkin (?). Mikäli yhdessä INPUT-lauseessa syötetään samalla kertaa useille muuttujille arvoja, voidaan syötettävät luvut ja/tai merkkijonot antaa yksi kerrallaan painamalla jokaisen luvun tai merkkijonon jälkeen ENTER:ä tai antamalla kaikki kerralla erottamalla syötettävät arvot pilkuilla. Jos kuvaruutuun tulostuu kaksi kysymysmerkkiä peräkkäin (??), tarkoittaa tämä sitä, että INPUT-lauseessa on pyydetty useampaa arvoa, kuin on syötetty. Suoritusta voidaan jatkaa syöttämällä sopiva luku/merkkijono ja painamalla ENTER.
Virheilmoitus "Redo from start" saadaan, jos syötettävät arvot eivät täsmää INPUT-lauseella pyydettäviin muuttujiin. (Esim. INPUT A# -lauseeseen yritetään syöttää merkkijonoa, vaikka pyydetään kaksinkertaisentarkkuuden lukua.) Suoritusta voidaan jatkaa syöttämällä oikeaa tyyppiä oleva arvo.
INPUT-lause voidaan pysäyttää kesken syötön painamalla CTRL+STOP -painikkeita. Tästä tilanteesta voidaan suoritusta jatkaa CONT-käskyllä.

ESIM:

```
10 PRINT "ANNA JOKIN LUKU"
20 INPUT A
30 PRINT
40 PRINT "ANTAMASI LUKU TOISEEN"
```

```
50 PRINT "TOISEEN KOROTETTUNA ON"; A*A
60 END
```

Rivillä 10 käytetään PRINT-käskyä tulostamaan tekstiä ja rivillä 20 ohjelman suoritus pysähtyy, tulostetaan kysymysmerkki, odotetaan jonkin luvun syöttämistä näppäimistöltä ja ENTER:n painamista. Jos yrität syöttää jonkin muun, kuin luvun (esim. merkkijonon), saat virheilmoituksen ja suoritus jatkuu vasta kun olet syöttänyt hyväksyttävän tiedon. Silloin suoritus jatkuu riviltä 30, jolla tulostetaan yksi tyhjä rivi. Rivit 40 ja 50 tulostavat tekstiä ja rivin 50 lopussa oleva yhtälö (A*A) antamasi luvun kerrottuna itsellään. Ohjelma loppuu riviin 60.
Rivit 10 ja 20 voidaan korvata yhdellä rivillä seuraavasti:

```
10 INPUT "ANNA JOKIN LUKU"; A
20 REM * TYHJÄ RIVI, JOKA JÄTETÄÄN KIRJOITTAMATTA *
```

Tässä käytetään INPUT-lauseen mahdollisuutta tulostaa tekstiä. Huomioi kuitenkin pieni eroavaisuus tulostuksessa. Nyt tulostuu kysymysmerkki tekstin "ANNA JOKIN LUKU" perään, kun edellisessä esimerkissä tulostui omalle riville. Rivi 20 jätetään kirjoittamatta.

```
10 INPUT "ANNA KAKSI LUKUA"; A,B
20 PRINT
30 PRINT "LUKUJEN TULO ON"; A*B
40 END
```

Tämä esimerkkiohjelma pyytää rivillä 10 syöttämään kaksi lukua, joiden tulo tulostetaan rivillä 30.

```
10 INPUT "ANNA NIMESI JA IKÄSI"; A$,B
20 PRINT
30 PRINT A$ ; ", OLET" ; B*365 ; "PÄIVÄÄ VANHA."
40 END
```

```
RUN <ENTER>
ANNA NIMESI JA IKÄSI? Ville <ENTER>
?? 14 <ENTER>

Ville, OLET 5110 PÄIVÄÄ VANHA.
Ok
```

Tällä kertaa pyydettiin syöttämään sekä merkkijono että numeerinen arvo. HUOM! Ne on syötettävä oikeassa järjestyksessä. Syötettävät tiedot voidaan antaa erikseen painamalla välillä ENTER:ä, kuten esimerkkiajossa tai antamalla molemmat yhdellä kertaa siten, että tiedot erotetaan pilkulla.

LET

MUOTO: [LET] <muuttuja> = <lause>

VERSIOT: MSX: KASET. + LEVYKE
SVI: KASET. + LEVYKE

TARKOITUS: Käskyllä annetaan muuttujalle <muuttuja> lauseen <lause> määräämä arvo.

LISÄYKSET: Poiketen muista käskyistä, voidaan itse käskysana LET jättää kirjoittamatta.
<muuttuja> = muuttuja, joka saa lauseen <lause> arvon. Voi olla minkä tyyppinen muuttuja tahansa.
<lause> = numero, merkkijono, matemaattinen tai merkkijonolause. Tulee vastata ilmaisun <muuttuja> tyyppiä. Merkkijonot erotetaan lainausmerkeillä.

ESIM:

```
10 LET A=3.14
20 LET B$="HARJOITUS"
30 LET C=10
40 LET D$="KERTO"
50 LET E$=D$+B$
60 LET F=A*C
70 PRINT D$
80 PRINT F
90 END
```

Riveillä 10—40 määritellään neljä muuttujaa, joista kaksi on numeerisia ja kaksi merkkijonoja. Rivillä 50 yhdistetään kaksi merkkijonomuuttujaa yhdeksi merkkijonoksi. Rivillä 60 saa uusi muuttuja F arvokseen kahden muuttujan (A ja C) tulon. Rivit 70 ja 80 ovat normaaleja PRINT-lauseita. Ohjelma päättyy riville 90 END-käskyyn.
Kokeile samaa ohjelmaa jättämällä riveiltä 10—60 LET-sanat pois.

LINE INPUT

MUOTO: **LINE INPUT ["<teksti>";] <muuttuja> [,<muuttuja>, . . .]**

VERSIOT: **MSX: KASET. + LEVYKE**
SVI: KASET. + LEVYKE

TARKOITUS: Käskyllä syötetään muuttujalle merkkijono.

LISÄYKSET: <teksti> = teksti, joka halutaan tulostaa ennen syöttöä.
<muuttuja> = muuttuja, jolle syötettävä merkkijono tallentuu.
Käsky toimii samoin kuin INPUT-käskykin. Erona vain on, että kysymysmerkkiä lauseen loppuun Ei tulosteta (ellei se ole osana ilmaisua <teksti>) ja että syötettävä merkkijono voi sisältää mitä tahansa merkkejä. Kaikki merkit ENTER:n painamiseen asti luetaan ilmaisuun <muuttuja> kuuluviksi.
LINE INPUT -käskyä on hyödyllistä käyttää silloin, kun merkkijonojen syöttö halutaan hoitaa käyttäjän tai pikemminkin ohjelmoijan mielen mukaisesti. Tällaisissa tapauksissa syötetystä merkkijonosta <muuttuja> poimitaan eri merkkijonokäskyillä halutut arvot tai muuttujat ja mahdollisten virheiden kohdatessa (merkkijonoja ja/tai numeerisia arvoja ei ole syötetty ohjelman vaatiman jatkon mukaisesti) siirrytään virheidenkäsittelyrutiiniin ja tarvittaessa pyydetään uudet arvot.

ESIM:

```
10 LINE INPUT "ANNA MERKKIJONO ? "; A$
20 PRINT A$
RUN <ENTER>
ANNA MERKKIJONO ? OLE HYVÄ "MIKRO";:("12., <ENTER>
OLE HYVÄ "MIKRO";:("12.,
Ok
```

Kokeile ohjelmaa syöttämällä kaikkia mahdollisia merkkejä. HUOM! maksimi merkkien määrä, jonka ilmaisu <muuttuja> voi saada on 254 merkkiä (vaatii tosin CLEAR-määrittelyn).

LIST

MUOTO: **LIST [<nro1>] [—<nro2>]**

VERSIOT: **MSX: KASET. + LEVYKE**
SVI: KASET. + LEVYKE

TARKOITUS: Käskyllä tulostetaan kuvaruudulle muistissa oleva ohjelma tai osa siitä.

LISÄYKSET: <nro1> = rivinumero, jolta tulostus alkaa.
<nro2> = rivinumero, johon tulostus loppuu.
Rivit tulostetaan aina alkaen pienimmästä rivinumerosta suurimpaan, huolimatta missä järjestyksessä rivit on talletettu. Pelkkä LIST-käsky tulostaa kaikki muistissa olevat rivit.

ESIM: Käskyllä voidaan tulostaa yksi rivi.

```
LIST <nro1>
LIST 55          ; tulostaa rivin 55.
LIST .           ; tulostaa viimeksi käsitellyn rivin (rivin, joka on viimeksi taltioitu, lisätty, korjattu, editoitu tai jossa on esiintynyt virhe).
```

Voidaan tulostaa rivit jostakin rivinumerosta lähtien.

```
LIST <nro>—
LIST 40—         ; tulostaa kaikki muistissa olevat rivit, alkaen riviltä 40.
LIST .—          ; tulostaa rivit, alkaen viimeksi käsitellyltä riviltä.
```

Voidaan tulostaa rivit johonkin rivinumeroon asti.

```
LIST —<nro2>
LIST —100        ; tulostaa rivit 0—100, mukaanlukien rivit 0 ja 100.
LIST —.          ; tulostaa rivit viimeksikäsitellylle riville asti, alkaen riviltä 0.
```

HUOM! vain rivit, joilla on ohjelmaa, tulostetaan, ei siis välttämättä riviä nolla (0) jne.
Voidaan tulostaa rivit joltakin väliltä.

```
LIST <nro1> — <nro2>
LIST 50—134      ; tulostaa rivit 50—134, mukaanlukien rivit 50 ja 134.
```

Jos annetaan esim. käsky LIST 19—49 ja muistissa on rivit 10—100 kymmenen välein, tulostuu rivit 20, 30 ja 40.

LIST-käskyn jälkeen siirrytään aina komentotilaan.
Jos haluat keskeyttää rivien tulostuksen näytöllä hetkeksi, paina STOP-näppäintä. Listaus jatkuu painamalla uudestaan STOP-näppäintä. Listaus voidaan keskeyttää kokonaan painamalla CTRL+STOP -näppäimiä.

LLIST

MUOTO: **LLIST [<nro1>] [—<nro2>]**

VERSIOT: **MSX: KASET. + LEVYKE**
SVI: KASET. + LEVYKE

TARKOITUS: Käskyllä tulostetaan muistissa oleva ohjelma tai osa siitä kirjoittimelle.

LISÄYKSET: <nro1> = rivinumero, jolta listaus aloitetaan.
<nro2> = rivinumero, johon listaus lopetetaan.
Käsky toimii samoin, kuin LIST-käskykin. Erona on, että tulostus tapahtuu kuvaruudun sijasta kirjoittimelle. Kts. LIST-käsky.
Kirjoittimen tulee olla oikein kytketty ja valmius tilassa (ON LINE). Varmistu, että paperi pääsee kulkemaan esteettä.
Ohjelman alkuun ja sekaan voidaan lisätä erillisiä kirjoittimen ohjauskomentoja, joilla saadaan tulostusta muunneltua (esim. lihavoitu, tummennettu teksti jne.). Kirjoitinohjauskomennot eivät vaikuta ohjelman suoritukseen. Ohjauskomennot löydät kirjoittimen mukana seuraavasta ohjekirjasta. Ohjauskomennot annetaan seuraavasti:

ESIM:
```
1  LPRINT CHR$ (XX)
2  LPRINT CHR$ (ZZ)
10 REM * OHJELMA ALKAA TÄSTÄ *
20 PRINT "MOI !"
     . . .
```

Riveillä 1 ja 2 on kirjoittimen ohjauskomento, jotka näkyvät vasta kirjoittimelle tulostuksen yhteydessä. Ilmaisut XX ja ZZ ovat jokin kokonaisluku (katso kirjoittimen ohjekirjasta).
Kokeile vastaavaa esimerkkiä LLIST-käskyllä, mikäli käytettävissäsi on kirjoitin.

LPRINT

MUOTO: **LPRINT [<lause>] [;<lause> ; . . .]**

VERSIOT: **MSX: KASET. + LEVYKE**
SVI: KASET. + LEVYKE

TARKOITUS: Käskyllä tulostetaan ilmaisun <lause> ilmoittama tieto kirjoittimelle.

LISÄYKSET: LPRINT-käsky toimii samoin, kuin PRINT-käskykin, mutta tulostus tapahtuu kirjoittimelle. Mikäli käytettävissäsi on kirjoitin, voit ajaa samat esimerkkiohjelmat lisäämällä jokaisen PRINT-käskyn eteen L-kirjaimen.
LPRINT-käskyä EI voi lyhentää L?, kun taas PRINT-käsky voidaan lyhentää kirjoitusvaiheessa ?-merkillä.
LPRINT-käskyllä voidaan kirjoittimelle antaa myös ohjauskomentoja, joilla tulostusasua saadaan muunneltua. Kirjoittimen ohjaus tapahtuu seuraavasti.

ESIM:
```
10 LPRINT CHR$ (XX)+CHR$ (ZZ)
20 LPRINT " TAULUKKO 1/1/1985 — 31/12/1985"
     . . .
```

Tässä esimerkissä annetaan rivillä 10 kirjoittimelle ohjauskomento, jonka jälkeen kirjoitin tulostaa esim. vahvennetulla tekstillä. Ilmaisujen XX ja ZZ tilalle tulee laittaa kirjoittimen hyväksymä ohjauskoodi (kts. ohjekirja).

LPRINT USING

MUOTO: **LPRINT USING "<muoto>";<lause> [; <lause>; . . .]**

VERSIOT: **MSX: KASET. + LEVYKE**
SVI: KASET. + LEVYKE

TARKOITUS: Käskyllä tulostetaan merkkijonoja halutussa muodossa kirjoittimelle.

LISÄYKSET: <muoto> = merkkijonovakio tai muuttuja, jolla määrätään tulostuksen muoto. Numeerisia lauseita ja merkkijonoja varten on omat tulostusohjaukset.
<lause> = lause, joka tulostetaan ilmaisun <muoto> määräämällä asulla.
Käsky toimii kuten PRINT USING -käskykin, erona tulostuksen siirtyminen kirjoittimelle. Katso PRINT USING -käsky.

ESIM:

```
10 A$="MATTI NYKÄNEN"
20 LPRINT USING "MÄKIHYPPÄÄJISTÄ PARAS ON ⓐ ";A$
30 LPRINT USING "HYVÄ \ \";A$
```

Jos Sinulla on kirjoitin, anna RUN-käsky ja saat tällaisen tulostuksen.

```
RUN <ENTER>
(kirjoitin)
MÄKIHYPPÄÄJISTÄ ON PARAS MATTI NYKÄNEN
HYVÄ MATTI
(kuvaruutu)
Ok
```

MID $

MUOTO: MID$(<string1>,<nro> [,<kpl>])=<string2>

VERSIOT: MSX: KASET. + LEVYKE
SVI: KASET. + LEVYKE

TARKOITUS: Käskyllä voidaan vaihtaa (korvata) ilmaisun <string1> merkeistä osa/kaikki uusiin ilmaisulla <string2> annettuihin merkkeihin.

LISÄYKSET: <string1> = merkkijono, josta osa/kaikki merkit korvataan uusilla merkeillä.
<nro> = vaihto voidaan aloittaa alkaen merkistä <nro> (vasemmalta lukien, kokonaisluku).
<kpl> = vaihdettavien merkkien määrä. Mikäli tätä ilmaisua ei anneta tai se on suurempi kuin luku, kuin ilmaisussa <string2> olevien merkkien määrä, vaihdetaan vain niin monta merkkiä, kuin alkuperäisessä merkkijonossa <string1> on. Ts. merkkijonon <string1> pituutta ei muuteta pidemmäksi, vaikka merkkijonossa <string2> olisi enemmän merkkejä.
<string2> = merkkijono, jolla merkkijonoa <string1> korvataan.

ESIM:

```
10 A$="MATTI PUSENIUS"
20 B$=A$
30 MID$(A$,7,7)="NYKÄNEN"
40 PRINT "NIMESTÄ ";B$
50 PRINT "TEHTIIN NIMI ";A$
60 END
```

```
RUN <ENTER>
NIMESTÄ MATTI PUSENIUS
TEHTIIN NIMI MATTI NYKÄNENS
Ok
```

Rivillä 10 määritellään merkkijonolle A nimi. Rivillä 20 kopioidaan merkkijono A merkkijonoksi B. Rivillä 30 vaihdetaan merkkijonosta A, alkaen merkistä "P" (7:s merkki), seitsemän merkkiä (merkit "NYKÄNEN"). Seuraavilla kahdella rivillä tulostetaan muunnos. Huomaa merkkijonoon A jäävä ylimääräinen "S"-kirjain. Tämän voi välttää muuttamalla rivin 30 käskyn seuraavasti:

```
30 MID$(A$,7,8)=" NYKÄNEN"
```

Nyt myös tulostus hieman muuttuu. Kokeile ja huomaa ero.

NEW

MUOTO: NEW

VERSIOT: MSX: KASET. + LEVYKE
SVI: KASET. + LEVYKE

TARKOITUS: Käskyllä poistetaan muistissa oleva ohjelma,-t ja kaikki muuttujat ja vakiot.

LISÄYKSET: Käskyn yhteydessä sulkeutuvat myös kaikki avoinna olevat tiedostot. NEW-käsky ei tuhoa (poista) ohjelman alussa CLEAR-käskyn yhteydessä määritellyn osoitteen yläpuolella olevia konekielisiä ohjelmia.
Käskyn suorituksen jälkeen palataan aina komentotilaan.

ESIM:

```
10  PRINT "ESIMERKKI"
20  REM POISTETAAN OHJELMA
30  REM MUISTISTA OHJELMALLISESTI
40  NEW
```

Aja ohjelma RUN-käskyllä ja anna sen jälkeen LIST-komento.
NEW-käsky ei varsinaisesti poista ohjelmaa,-ia muistista, mutta hukkaa tärkeän tiedon kahdesta muistipaikasta ja tekee näin esim. listauksen mahdottomaksi käskyn jälkeen. Palauttamalla POKE-käskyllä oikeat tiedot oikeisiin muistipaikkoihin saadaan listaus taas näkyviin.

ON ERROR GOTO

MUOTO: ON ERROR GOTO <nro>

VERSIOT: MSX: KASET. + LEVYKE
SVI: KASET. + LEVYKE

TARKOITUS: Käskyllä määrätään rivinumero, jolta suoritus jatkuu virheen tapahduttua.

LISÄYKSET: <nro> = rivinumero, jolle suoritus siirtyy virheen tapahduttua.
Kun ohjelmassa on annettu tämä määrittely (usein ohjelman alussa), aiheuttavat kaikki virheet hypyn ilmaisulla <nro> määritellylle riville. Määrittely on voimassa, kunnes annetaan ON ERROR GOTO 0 -käsky. Tämän jälkeen virheen tapahduttua annetaan normaali virheilmoitus ja siirrytään komentotilaan. Näin tapahtuu myös silloin, jos itse virheenkäsittelyrutiinissa tapahtuu virhe.

ESIM:

```
10  ON ERROR GOTO 100
20  PRINT "TÄMÄ OHJELMA LASKEE NELIÖJUURIA"
30  PRINT "HALUAMALLESI LUVULLE. TESTAAMME"
40  PRINT "VIRHERUTIINEJA ANTAMALLA NEGA-"
50  PRINT "TIIVISEN JUUREN. ALOITETAAN. . ."
60  PRINT
70  INPUT "ANNA JOKIN LUKU "; A
80  PRINT "NELIÖJUURI LUVUSTA ON"; SQR(A)
90  PRINT "JA UUDESTAAN . . ." : GOTO 60
100 REM * VIRHERUTIINI *
110 PRINT
120 PRINT "     OLEMME VIRHERUTIINISSA."
130 PRINT "  JOS HALUAT KORJATA VIRHEESI,"
140 INPUT "PAINA Y JA ENTER, MUUTEN JOKU MUU";A$
150 IF A$="Y" OR A$="y" THEN RESUME 60
160 PRINT "OKEI, EI KORJATA, VAAN"
170 PRINT "PALATAAN KOMENTOTILAAN."
180 ON ERROR GOTO 0
```

Rivillä 10 määritellään virherutiini alkavaksi riviltä 100. Rivit 20—50 tulostavat ohjeita ohjelmasta. Rivillä 60 tulostetaan tyhjä rivi. Ohjelmasilmukan eli loopin muodostavat rivit 60—90, joilla pyydetään käyttäjältä jotain lukua INPUT-lauseella. Jos annetaan positiivinen luku, tulostuu luvusta neliöjuuri ja siirrytään odottamaan uutta lukua. Jos syötetään negatiivinen luku, siirtyy ohjelma riville 100. Tämä aliohjelma kysyy rivillä 140, halutaanko tehty virhe korjata. Jos halutaan, siirtyy suoritus rivillä 140 olevan RESUME 60 -käskyn ansiosta riville 60 (kts. RESUME-käsky). Jos tehtyä virhettä ei haluta korjata ja vastataan jotain muuta, palaa suoritus komentotilaan rivillä 180 olevan määrittelyn takia. Tällöin tulostuu myös normaali virheilmoitus "Illegal function call in 80".

ON GOTO

MUOTO: ON <lause> GOTO <nro1> [,<nro2> , . . .]

VERSIOT: MSX: KASET. + LEVYKE
SVI: KASET. + LEVYKE

TARKOITUS: Käskyllä ohjataan suoritus riville/riveille, jotka luetellaan halutussa järjestyksessä.

LISÄYKSET: <lause> = numero tai useimmin matemaattinen lause, jonka arvo lasketaan ja suoritetaan hyppy vastaavalle riville, joiden luettelo seuraa GOTO-sanaa.

<nro1> = rivinumero, jolle suoritus siirtyy, mikäli ilmaisun <lause> arvo on yksi.
<nro2> = rivinumero, jolle suoritus siirtyy, mikäli ilmaisun <lause> arvo on kaksi jne.
Jos rivinumeroiden luettelosta <nro1>, <nro2>, . . . ei löydy ilmaisulle <lause> vastaavaa numeroa, jatkuu suoritus ON GOTO -lausetta seuraavalta riviltä (tai käskystä). Ilmaisun <lause> saadessa desimaaliarvon, suoritetaan pyöristys lähinnä pienempään kokonaislukuun.
Jos ilmaisu <lause> saa arvon nolla (0), siirrytään seuraavalle riville tai suoritetaan seuraava käsky. Mikäli <lause>:n arvo ei ole välillä 1—255, tulostetaan virheilmoitus "Illegal function call".

ESIM:

```
10  INPUT "ANNA JOKIN LUKU VÄLILTÄ 1—4";A
20  ON A GOTO 50,70,90,110
30  PRINT "ET ANTANUT OIKEAA LUKUA, ANNA UUDESTAAN."
40  GOTO 10
50  PRINT "ANNOIT LUVUN 1."
60  GOTO 10
70  PRINT "ANNOIT LUVUN 2."
80  GOTO 10
90  PRINT "ANNOIT LUVUN 3."
100 GOTO 10
110 PRINT "ANNOIT LUVUN 4."
120 PRINT "KIITOS SEURASTA !"
130 END
```

Ohjelma pyytää rivillä 10 syöttämään jotain lukua väliltä 1—4 (HUOM! myös desimaaliluku käy). Rivillä 20 suoritetaan hyppy annetun luvun perusteella (jos esim. annettiin luku 3, siirrytään riville, jonka numero on kolmantena luettelossa). Ohjelma päättyy luvulla neljä (4).
Ilmaisut <nro1>, <nro2>, jne. eivät tarvitse olla suuruusjärjestyksessä.

ON GOSUB

MUOTO: ON <lause> GOSUB <nro1> [,<nro2>, . . .]

VERSIOT: MSX: KASET. + LEVYKE
SVI: KASET. + LEVYKE

TARKOITUS: Käskyllä ohjataan suoritus aliohjelmaan/aliohjelmiin riveille, joiden luettelo seuraa GOSUB-sanaa.

LISÄYKSET: <lause> = numero tai usemmin matemaattinen lause, jonka arvo lasketaan ja suoritetaan hyppy aliohjelmaan, joiden luettelo annetaan halutussa järjestyksessä.
<nro1> = rivinumero, jolta alkaa aliohjelma numero yksi.
<nro2> = rivinumero, jolta alkaa aliohjelma numero kaksi.
ON GOSUB -lauseelle pätee samat ehdot, kuin ON GOTO -lauseellekkin. Paluu aliohjelmasta tapahtuu RETURN-käskyllä seuraavalle riville tai käskyyn.

ESIM:

```
10  INPUT "ANNA JOKIN LUKU VÄLILTÄ 1—4";A
20  ON A GOSUB 70,50,110,90
30  PRINT "KIITOS SEURASTA."
40  END
50  PRINT "ANNOIT LUVUN 2"
60  RETURN
70  PRINT "ANNOIT LUVUN 1"
80  RETURN
90  PRINT "ANNOIT LUVUN 4"
100 RETURN
110 PRINT "ANNOIT LUVUN 3
120 RETURN
```

Rivillä 10 pyydetään lukua väliltä 1—4. Rivillä 20 suoritetaan hyppy aliohjelmaan annetun luvun perusteella. HUOM! rivinumeroiden järjestys. Paluu aliohjelmista suoritetaan RETURN-käskyllä (vrt. rivit 60,80,100 ja 120).

PRINT

MUOTO: **PRINT** [<lauseita>]

VERSIOT: **MSX: KASET. + LEVYKE**
SVI: KASET. + LEVYKE

TARKOITUS: Käskyllä tulostetaan näytölle vakioita, numeerisia- ja merkkijonomuuttujia, tekstiä, jne.

LISÄYKSET: PRINT-lauseita koskevat seuraavat säännöt:

* lainausmerkeissä oleva teksti tulostetaan sellaisenaan.
* merkkijonojen eteen tai jälkeen ei tulosteta välilyöntejä (tyhjää).
* kaikkia lukuja seuraa yksi välilyönti (tyhjä).
* positiivisia lukuja edeltää yksi välilyönti (+ merkki korvataan tyhjällä).
* negatiivisia lukuja edeltää miinus (—) -merkki.
* jos lauseiden välissä on pilkku (,), tulostuu jälkimmäinen lauseke alkaen seuraavasta tabulaattoripisteestä jne.
* jos lausetta seuraa puolipiste (;), estyy automaattinen rivitabulointi ja lauseke tulostuu edellisen perään.
* matemaattisten lauseiden arvot lasketaan ennen tulostusta.
* jos viimeistä lausetta ei seuraa puolipiste tai pilkku, siirtyy kohdistin seuraavan rivin alkuun.
* jos tulostettava lause on pidempi kuin rivin pituus, siirtyy kohdistin (=kursori) rivin tullessa täyteen seuraavan rivin alkuun, josta tulostus jatkuu.
* tyhjä rivi saadaan pelkällä PRINT-käskyllä.
* tabulointiväli on; MSX: 8 merkkiä.
  SVI: 8 merkkiä.

PRINT-käsky voidaan editointi vaiheessa korvata kysymysmerkillä (?), jonka BASIC muuttaa PRINT-sanaksi.

ESIM:

```
PRINT "17+17=";17+17 <ENTER>
17+17= 34
Ok

? "TEKSTIÄ JA ERIKOISMERKKEJÄ = ; : , . / ?" <ENTER>
TEKSTIÄ JA ERIKOISMERKKEJÄ = ; : , . / ?
Ok

10   A$="HARJOITUS"
20   ? A$ , A$ ; A$
LIST <ENTER>
10   A$="HARJOITUS"
20   PRINT A$ , A$ ; A$
Ok
RUN <ENTER>
HARJOITUS      HARJOITUSHARJOITUS
Ok

10   A$="TESTI"
20   PRINT A$ , , A$
RUN <ENTER>
TESTI
TESTI
Ok

10   INPUT "ANNA KAKSI LUKUA "; A,B
20   PRINT "LUKUJEN SUMMA ON"; A+B
30   PRINT "JA TULO ON"; A*B
RUN <ENTER>
ANNA KAKSI LUKUA ? 3,7 <ENTER>
LUKUJEN SUMMA ON 10
JA TULO ON 21
Ok
```

PRINT TAB -käskyllä voidaan tulostus siirtää haluttuun kohtaan rivillä (kts. TAB-käsky). LOCATE-käsky siirtää kursorin annettujen kordinaattien määräämään kohtaan, josta voidaan tulostusta jatkaa PRINT-käskyllä (kts. LOCATE-käsky).
MSX-BASIC:ssä voidaan PRINT-käskyllä tulostaa SCREEN-moodeissa nolla (0) ja yksi (1) eli SCREEN 0 ja SCREEN 1 ilman lisätoimenpiteitä. SCREEN-moodit 2 ja 3 vaativat ennen tulostusta OPEN-komennon, jolla avataan tiedosto tulostettavaa tietoa varten.
SVI-BASIC:ssä voidaan PRINT-käskyä käyttää kaikissa SCREEN-moodeissa ilman lisätoimenpiteitä.

PRINT USING

MUOTO: PRINT USING "<muoto>";<lause>

VERSIOT: MSX: KASET. + LEVYKE
SVI: KASET. + LEVYKE

TARKOITUS: Käskyllä tulostetaan numeroita tai merkkijonoja halutussa muodossa.

LISÄYKSET: <muoto> = ilmaisulla annetaan muoto, jolla tulostus suoritetaan. Numeerisia arvoja ja merkkijonoja varten on omat muodot, jotka esitetään jäljempänä.
<lause> = tulostettava lause. Mikäli lauseita on useita, erotetaan ne toisistaan joko pilkulla tai puolipisteellä.
Numeerisia arvoja ja merkkijonoja varten on omat muodot, jotka ovat seuraavat:

A. MERKKIJONOMUUTTUJAT

"!" ; vain merkkijonon ensimmäinen merkki tulostetaan.

ESIM:

```
10  A$="harjoitus"
20  PRINT USING "TÄMÄ ON !elppoa"; A$

RUN <ENTER>
 TÄMÄ ON helppoa
Ok
```

"\X kpl tyhjiä\" ; tulostetaan 2+X kpl merkkejä halutusta merkkijonosta. Jos välilyöntejä ei anneta, tulostetaan 2 merkkiä, jos annetaan 1 välilyönti, tulostuu 3 merkkiä, jne.

ESIM:

```
10  A$="HARJOITUS"
20  PRINT USING "TÄMÄ ON\   \A"; A$

RUN <ENTER>
TÄMÄ ON HARJA
Ok
```

Jos merkkijono on lyhyempi, kuin sille varattu tila, tulostuu merkkijonon perään välilyöntejä niin paljon, että varattu tila tulee täyteen.

"ⓐ" ; merkkijono tulostuu kokonaan huolimatta pituudesta.

ESIM:

```
10  A$="HARJOITUS"
20  PRINT USING " ⓐ ON HYÖDYLLISTÄ. "; A$

RUN <ENTER>
HARJOITUS ON HYÖDYLLISTÄ.
Ok
```

B. NUMEERISET LAUSEET

"#" ; merkillä määrätään tulostettavien merkkien määrä. Jokaista #-merkkiä vastaa yksi numero. Jos tulostettava luku ei täytä sille varattua tilaa, tulostuu sen eteen välilyöntejä.

ESIM:

```
10  A=SQR(3)
20  PRINT USING "# # #.# # # #"; A , SQR(8) , 7 ,
2.55555 , 1234

RUN <ENTER>
 1.7321  2.8284  7.0000  2.5556%1234.0000
Ok
```

Desimaalipiste voidaan sijoittaa minne tahansa kentässä (vrt. esimerkki). Jos desimaalipistettä käytetään tulostuu luku aina samalla tarkkuudella, jopa nollat (0) tulostuu. Tarvittaessa suoritetaan pyöristys. Jos tulostettava luku on suurempi, kuin annettu muoto, tulostuu luvun eteen prosentti (%) -merkki.

"+" ; merkki määrää tulostettavaksi luvun etumerkin (+ tai —). Jos plus-merkki on sijoitettu luvun eteen, tulostuu etumerkki luvun eteen ja perään sijoitettuna päinvastoin.

ESIM:

```
10  A=SQR(5)
20  PRINT USING "+ # #.# #"; A , 2.55555
30  PRINT USING "# #.# # # #+; SQR(7) , 7

RUN <ENTER>
 +2.24 +2.56
 2.65+ 7.00+
Ok
```

"—" ; miinus (—) -merkki määrittelyn jäljessä tulostaa negatiivisilla luvuilla miinus-merkin luvun perään ja positiivisilla luvuilla tulostuu yksi välilyönti.

ESIM:

```
10   A=SQR(2)
20   PRINT USING "# #.# #—"; A , 2—13

RUN <ENTER>
 1.41 11.00—
Ok
```

"**" ; merkinnällä tulostetaan tyhjiin paikkoihin (eli luku ei täytä sille varattua tilaa) tähti. Samalla tulee tilaa kahdelle lisänumerolle.

ESIM:

```
10   A=SQR(2)
20   PRINT USING "** # #.# #"; A , 2—13
30   PRINT USING "# #.# #**"; SQR(7) , 7

RUN <ENTER>
***1.41*—11.00
 2.65*7
Ok
```

"," ; pilkku ennen desimaalipistettä tulostaa pilkun joka kolmannen numeron jälkeen. Pilkku määrittelyosan lopussa katsotaan osaksi lukua. Pilkku lisää yhden merkin tulostukseen. Jos luvussa käytetään eksponenttiesitystä, esitystä, ei pilkun käytöllä ole vaikutusta.

ESIM:

```
10   A=SQR(2)
20   PRINT USING "# # # # #,.# #"; A , 10 *123.45
30   PRINT USING "# # # # #.# #,"; A , 10*123.45

RUN <ENTER>
 1.41 1,234.50
 1.41, 1234.50,
Ok
```

"^^^^" ; eksponenttimuodon määritys. Määritys tekee tilaa merkinnälle 'E+xx', jossa E-kirjain kuvaa eksponenttimuotoa ja xx on eksponentti. Desimaalipisteen käyttö on sallittua. Myös plus- ja miinusmerkki ovat sallittuja.

ESIM:

```
10   A=SQR(2)
20   PRINT USING "# #.# #^^^^"; A , 123.45 , —678.91
30   PRINT USING "#.# # #^^^^—"; A, 123.45 , —678.91
40   PRINT USING "+ #.# #^^^^"; A , 123.45 , —678.91

RUN <ENTER>
 1.41E+00 1.23E+02—6.79E+02
1.414E+00 1.235E+02 6.789E+02—
+1.41E+00+1.23E+02—6.79E+02
```

"$$" ; ilmaisulla tulostetaan luvun eteen dollari ($) -merkki. Ilmaisu tekee samalla tilaa kahdelle merkille, joista toinen on $-merkki. Eksponenttimuodon käyttö ei ole sallittua. Jos ilmaisun perään lisätään miinus (—) -merkki, tulostuu negatiivisen luvun perään miinus.

ESIM:

```
10   A=SQR(2)
20   PRINT USING "$$# #.# #"; A , —33.55
30   PRINT USING "$$# #.# #—"; A , —33.55

RUN <ENTER>
 $1.41—$33.55
 $1.41 $33.55—
Ok
```

"**$" ; ilmaisu tekee tilaa kolmelle merkille, joita yksi on lukua edeltävä dollarimerkki ja loput lukua edeltävät tyhjät korvataan tähdillä (*). Jos luku on negatiivinen, tulostuu dollarimerkin eteen miinus (—) -merkki.

ESIM:

```
10   A=SQR(27)
20   PRINT USING "**$# #.# #"; A , —33.55
30   PRINT USING "**$# #.# #—"; A, —33.55
```

```
RUN <ENTER>
***$1.41*—$33.55
***$1.41 **$33.55—
Ok
```

Jos tulostettava lause tai luku on pidempi, kuin sille varattu tila, tulostuu kaikilla muodoilla luvun eteen %-merkki. Samoin käy, jos luku joudutaan pyöristämään kenttämäärittelyn vuoksi.

ESIM:

```
10  A=123.45
20  PRINT USING "# #.# #"; A
30  PRINT USING ".# #"; 1.999

RUN <ENTER>
%123.45
%2.00
Ok
```

READ

MUOTO: READ <muuttuja> [,<muuttuja>, . . .]

VERSIOT: MSX: KASET. + LEVYKE
SVI: KASET. + LEVYKE

TARKOITUS: Käskyllä luetaan tietoa DATA-lauseilta. Luetut tiedot sijoitetaan READ-lauseella annettujen muuttujien arvoiksi.

LISÄYKSET: <muuttuja> = muuttuja, joka saa sisällökseen DATA-lauseelta luetun arvon.
Ohjelman ensimmäinen READ-käsky lukee tiedon muuttujalle ensimmäiseltä DATA-lauseelta. Jos ohjelmassa on yksikin READ-käsky, täytyy ohjelmassa olla vähintään yksi DATA-lause. DATA-lauseita saa olla enemmän kuin READ-lauseita.
Jos READ-käsky annetaan ja muuttujalle ei löydetä vapaata dataa (kaikki datat luettu tai datoja on liian vähän), tulostuu virheilmoitus "Out of DATA".
Aina, kun tietoa READ-käskyllä luetaan, kasvatetaan nk. datalaskuria, joka pitää huolen siitä, mitä tietoa ja miltä DATA-lauseelta seuraavaksi luetaan. BASIC huolehtii lukujärjestyksestä, mutta järjestystä voidaan ohjelmallisesti muuttaa (kts. RESTORE-käsky).
Yhdellä READ-käskyllä voidaan lukea yksi tai useampi data, jolloin muuttujat erotetaan pilkulla. Muuttujien on vastattava DATA-lauseen vakioita tai tulostuu virheilmoitus "Syntax error".
DATA-lauseiden vakiot voivat olla numeeria tai merkkijonoja. Mikäli merkkijonot sisältävät erikoismerkkejä, kuten pilkku (,), kaksoispiste (:), jne., tulee merkkijonot erottaa lainausmerkeillä.

ESIM:

```
10  CLS
20  PRINT "        NIMI" ; "          PISTEET"
30  PRINT
40  FOR A%=1 TO 4
50  READ A$(A%) , B%(A%)
60  PRINT A$(A%) ; TAB(22); B%(A%)
70  NEXT
80  DATA "1. MATTI NYKÄNEN" , 245.5
90  DATA "2. JARI PUIKKONEN" , 239.8 , "3. PENTTI KOKKONEN"
100 DATA 232.4
110 DATA "4. JENS WEISFLOG"
120 DATA 227.7
```

Rivillä 10 tyhjennetään näyttö. Rivillä 20 tulostetaan otsikot ja rivillä 30 yksi tyhjä rivi. Rivit 40—70 muodostavat FOR-NEXT -silmukan. Rivillä 50 luetaan kaksi dataa, joista ensimmäinen on merkkijono ja toinen numeerinen vakio. Rivillä 60 tulostetaan luetut tiedot; ensin merkkijono (nimi) ja sitten numero (pisteet). Pisteet tulostetaan TAB-käskyn avulla sarakkeeseen 22. Rivillä 70 oleva NEXT-käsky sulkee silmukan. Rivit 80—120 sisältävät tiedot, jotka READ-käskyllä luetaan. Huomioi, että yhdellä DATA-rivillä voi olla yksi tai useampi vakio.

REM

MUOTO: REM [<teksti>]

VERSIOT: MSX: KASET. + LEVYKE
SVI: KASET. + LEVYKE

TARKOITUS: Käskyllä lisätään ohjelmaan selventäviä huomautuksia.

LISÄYKSET: <teksti> = mitä tahansa tekstiä, joka näkyy vain listauksessa. Voi sisältää kaikkija merkkejä.
REM-lauseita EI ohjelman ajon aikana suoriteta. Ne näkyvät vain LIST-käskyllä. Lauseilla ilmoitetaan yleensä ohjelman nimi, tekijä, annetaan selventäviä huomautuksia myöhempää ohjelman korjausta/muuntoa varten, jne.
Jos rivillä on muitakin käskyjä, on REM-lause sijoitettava viimeiseksi. REM-lauseet erotetaan muista käskyistä normaalisti kaksoispisteellä (:).
REM-käsky voidaan korvata heittomerkillä ('), jolloin kaksoispistettä EI tarvita erottamaan REM-lausetta toisista lauseista, joiden jäljessä REM-lause on.
Riville, jolla on REM-lause, voidaan hypätä esim. GOTO- ja GOSUB-käskyillä, jolloin siirrytään seuraavalle riville, jolla on suoritettava käsky. REM-lauseita EI voi käyttää DATA-lauseissa.

ESIM:

```
10  REM ******************************
20  REM ****** REM-ESIMERKKI ******
30  REM ******************************
40  '
50  '
60  INPUT "MIKÄ SINUN NIMI ON "; A$ : REM KYSYTÄÄN NIMI
70  PRINT "TERVE, "; A$ ' TULOSTETAAN TERVE, NIMI
80  '
90  REM ******************************
100 REM *********  LOPPU  *********
110 REM ******************************
```

Rivit 10—30 ovat REM-lauseet, joilla tulostetaan ohjelman tarkoitus. Rivit 40 ja 50 ovat myös REM-lauseita. Rivillä 60 pyydetään käyttäjää syöttämään nimi INPUT-lauseella ja REM-lause on lisätty rivin loppuun selventämään INPUT-rivin tarkoitusta. Rivillä 70 tulostetaan tervehdys ja lisänä on REM-lause, joka on merkitty heittomerkillä. HUOM! kaksoispistettä ei tarvita erottamaan REM-lausetta PRINT-lauseesta. Rivit 80—110 ovat REM-lauseita.

RENUM

MUOTO: RENUM [<uusi>] [,<vanha>] [,<lisäys>]

VERSIOT: MSX: KASET. + LEVYKE
SVI: KASET. + LEVYKE

TARKOITUS: Käskyllä uudelleennumeroidaan ohjelmarivit.

LISÄYKSET: <uusi> = rivinumero, joka tulee ensimmäiseksi uudeksi rivinumeroksi.
<vanha> = vanha rivinumero, jolta uudelleennumerointi alkaa.
<lisäys> = haluttu rivinumeroiden väli. Oletusarvo 10.
Jos ilmaisu <uusi> jätetään antamatta, aloitetaan rivinumerointi numerosta kymmenen (10).
Jos ilmaisu <vanha> jätetään antamatta, aloitetaan rivinumerointi ensimmäiseltä muistissa olevalta riviltä.
Käsky muuttaa myös GOTO-, GOSUB-, IF—THEN—ELSE-, ERL- jne. käskyjen rivinumerot suhteessa vastaaviksi. Jos näiden käskyjen rivinumeroista löytyy rivinumero, jolla ei ole ohjelmaa, tulostetaan virheilmoitus "Undefined line XXXXX in YYYYY". Numero XXXXX on rivinumero, jota ei ole olemassa tai jolla ei ole ohjelmaa. Numero YYYYY on rivinumero, jolla virhe esiintyi. Tässä tapauksessa RENUM-käsky ei muuta numeroa XXXXX, mutta muuttaa tarvittaessa numeron YYYYY.

ESIM:

KÄSKY	VANHAT RIVINUM.	UUDET RIVINUM.
RENUM	10,20,30,40,...	10,20,30,40,...
RENUM ,1	10,20,30,40,...	10,20,30,40,...
RENUM ,,1	10,20,30,40,...	10,11,12,13,...
RENUM 2,,3	10,20,30,40,...	2,5,8,11,14,...
RENUM 70,20	10,20,30,40,...	10,70,80,90,...
RENUM 60,30,5	10,20,30,40,...	10,20,60,65,...

Käskyllä EI voi muuttaa rivien järjestystä.

ESIM:

```
10   REM HARJOITUS
20   PRINT "RIVI 20"
30   PRINT "RIVI 30"
40   PRINT "RIVI 40"

RENUM 25,40 <ENTER>
Illegal function call
Ok
```

Käskyä RENUM 25,40 ei suoriteta, koska riviä 40 ei voida sijoittaa ennen riviä 30. Huomaa, että käsky ei muuta rivien numeroita.

RESTORE

MUOTO: RESTORE [<nro>]

VERSIOT: MSX: KASET. + LEVYKE
SVI: KASET. + LEVYKE

TARKOITUS: Käskyllä sallitaan kertaalleen luettujen DATA-lauseiden luku uudestaan.

LISÄYKSET: <nro> = uudelleenluku voidaan aloittaa ilmaisulla <nro> ilmoitettavalta riviltä. RESTORE-käsky siirtää datalaskurin osoittimen ensimmäiseen DATA-vakioon ja RESTORE <nro> -käsky rivillä <nro> olevaan ensimmäiseen data-alkioon.

ESIM:

```
10   PRINT "NIMI          IKÄ"
20   PRINT
30   FOR A%=1 TO 3
40   READ B$ , C
50   RESTORE 90
60   PRINT B$ , C
70   NEXT
80   DATA HEIKKI , 15
90   DATA TEIJA , 22
100  DATA WILLE , 31

RUN <ENTER>
 NIMI        IKÄ
HEIKKI       15
TEIJA        22
TEIJA        22
Ok
```

Rivillä 10 tulostetaan otsikot ja rivillä 20 yksi tyhjä rivi. Rivit 30—70 muodostaa FOR-NEXT -silmukan, jonka aikana ohjelma lukee muuttujia READ-käskyllä (rivi 40), antaa käskyn käyttää datoja riviltä 90 (HUOM! rivin 90 datoja ei ole vielä luettu, mutta voidaan silti antaa RESTORE 90 -käsky) rivillä 50 ja tulostaa luetut datat rivillä 60. Rivin 70 NEXT-käsky sulkee silmukan. Rivit 80 ja 90 sisältävät ohjelman käyttämiä datoja. Rivin 100 datoja ei ohjelma käytä.

RESUME

MUOTO: RESUME [<lause>]

VERSIOT: MSX: KASET. + LEVYKE
SVI: KASET. + LEVYKE

TARKOITUS: Käskyllä osoitetaan rivinumero, jolta virheenkäsittelyrutiinin jälkeen jatketaan ohjelman suoritusta.

LISÄYKSET: Ilmaisu <lause> voi olla jokin seuraavista.

```
RESUME          ; suoritus jatkuu käskystä, jolla virhe tapahtui.
RESUME 0        ; sama kuin edellä.
RESUME NEXT     ; suoritus jatkuu virhettä seuraavasta käskystä.
RESUME XXXXX    ; suoritus jatkuu riviltä numero XXXXX (muu kuin nolla).
```

Virheenkäsittelyrutiinista on palattava aina RESUME-käskyllä, eikä esim. GOTO-käskyllä.

ESIM:

```
10  ON ERROR GOTO 100
20  REM  PYYDETÄÄN INPUT-KÄSKYLLÄ
30  REM  LUKUA JA KUN NEGATIIVINEN
40  REM  LUKU SYÖTETÄÄN, TAPAHTUU
50  REM  RIVILLÄ 80 VIRHE (NELIÖ-
60  REM  JUURI NEGAT. LUVUSTA).
70  INPUT "ANNA JOKIN LUKU "; A
80  PRINT "LUVUN NELIÖJUURI ON "; SQR(A)
90  GOTO 70
100 '
110 PRINT
120 PRINT "VIRHE"
130 PRINT "ANNA UUSI LUKU "; A
140 RESUME
```

Rivillä 10 määritellään rivinumero (100), jolle suoritus siirtyy virheen tapahtuessa. Rivit 20—60 ovat selventäviä REM-lauseita, jotka eivät ohjelman suoritukseen vaikuta. Rivillä 70 pyydetään käyttäjää syöttämään jokin luku ja jos syötetään negatiivinen luku, tapahtuu seuraavalla rivillä (80) virhe, kun negatiivisesta luvusta yritetään ottaa neliöjuurta. Tällöin suoritus siirtyy alussa ON ERROR GOTO -lauseella määritellylle riville (rivi 100), jolta suoritusta jatketaan. Rivillä 100 on REM-käsky korvattu heittomerkillä (kts. REM-käsky). Rivillä 110 tulostetaan tyhjä rivi. Rivi 120 tulostaa tekstin "VIRHE". Rivillä 130 pyydetään uutta lukua ja kun se annetaan, siirrytään riville 140, jolla on paluukäsky RESUME. Paluu tapahtuu riville 80.

RUN

MUOTO: **RUN [<nro>]**

VERSIOT:
MSX: KASET. + LEVYKE
SVI: KASET. + LEVYKE

TARKOITUS: Käskyllä käynnistetään muistissa oleva ohjelma.

LISÄYKSET: <nro> = rivinumero, jolta ohjelman suoritus aloitetaan.
Jos ilmaisua <nro> ei anneta, käynnistyy muistissa oleva ohjelma sen ensimmäiseltä riviltä, muussa tapauksessa annetulta riviltä.

ESIM:

```
KÄSKY          KÄYNNISTÄÄ
RUN       ; muistissa olevan ohjelman ensimmäiseltä riviltä.
RUN 120   ; muistissa olevan ohjelman riviltä numero 120. Muuttujat nolla-
            taan.
```

RUN

MUOTO: **RUN "[<dev>:] <nimi>" [,R]**

VERSIOT:
MSX: KASET. + LEVYKE
SVI: KASET. + LEVYKE

TARKOITUS: Käskyllä ladataan ohjelma muistiin, jonka jälkeen ohjelma käynnistetään.

LISÄYKSET: <dev> = laite, jolta ohjelma ladataan. Voi olla jokin seuraavista:

```
LAITE          TARKOITTAA
1         ; levykeasema yksi (1). SVI DISK BASIC.
2         ; levykeasema kaksi (2). SVI DISK BASIC.
A         ; levykeasema yksi (1). MSX DISK BASIC.
B         ; levykeasema kaksi (2). MSX DISK BASIC.
CAS       ; nauha-asema.
```

Jos ilmaisu <dev> jätetään antamatta, olettaa MSX-BASIC (DISK-versio) laitteeksi levykeaseman ja muuten kasettiaseman. SVI-BASIC olettaa tällöin laitteeksi nauha-aseman (molemmat versiot).
<nimi> = ladattavan/käynnistettävän ohjelman nimi. Voi MSX DISK BASIC:ssä sisältää kahdeksan (8) merkkiä nimiosassa ja kolme (3) merkkiä tarkennusosassa, välissä piste. SVI-BASIC:n molemmissa versioissa voi nimiosa sisältää kuusi (6) merkkiä ja tarkennusosa kolme (3) merkkiä.

Jos ilmaisu ",R" liitetään käskyn perään, pidetään kaikki avoinna olevat datatiedostot auki myöhempää lukua/kirjoitusta varten.
RUN-käsky poistaa kaiken muistissa olevan tiedon ladatessaan uutta ohjelmaa.

ESIM:

KÄSKY	LATAA
RUN "demo"	; ohjelman nimeltä "demo" (lataus nauhurilta). Latauksen päätyttyä käynnistyy ohjelma pienimmältä riviltä.
RUN "1:GAME1",R	; ohjelman "GAME1" (lataus levykeasemasta 1). Avoinna olevat tiedostot jätetään ladattavan aliohjelman käyttöön. Käsky toimii vain SVI DISK BASIC:ssä.

STOP

MUOTO: STOP

VERSIOT: MSX: KASET. + LEVYKE
SVI: KASET. + LEVYKE

TARKOITUS: Käskyllä keskeytetään ohjelman suoritus.

LISÄYKSET: STOP-käskyä käytetään virheiden etsinnässä ja ohjelman testausvaiheessa. Kun käskyllä pysäytetään ohjelman suoritus, ei muuttujia nollata ja tiedostot jätetään avoimiksi. Samalla suoritus siirtyy komentotilaan ja esim. muuttujien arvoja voidaan PRINT-käskyllä tutkia. Kun ohjelma on testattu, poistetaan STOP-käskyt tai korvataan ne REM-käskyillä. Vrt. END-käsky.
STOP—käsky tulostaa ilmoituksen "Break in XXXXX", jossa XXXXX on rivinumero, jolla käsky kohdattiin. Ohjelman suoritusta voidaan tällöin jatkaa joko CONT-käskyllä tai GOTO <rivinumero> -käskyllä.
Jos ohjelmaa muutetaan Break-tilanteen jälkeen, ei CONT-käskyä voida antaa, vaan silloin on käytettävä GOTO-käskyä.

ESIM:

```
10   PRINT "          STOP-KÄSKYN HARJOITUS"
20   SAT=RND(1)
30   REM  TEHDÄÄN BREAK
40   STOP
50   REM  CONT-KÄSKYN JÄLKEEN JATKUU TÄLTÄ RIVILTÄ
60   PRINT 100*SAT
70   END

RUN <ENTER>
 STOP-KÄSKYN HARJOITUS
Break in 40
Ok
PRINT SAT <ENTER>
 .59521943994623
Ok
CONT <ENTER>
 59.521943994623
Ok
```

Rivillä 10 tulostetaan otsikko. Rivillä 20 arvotaan muuttujalle SAT satunnaisluku RND-käskyllä. REM-lauseella rivillä 30 tulostetaan kommentti. Rivillä 40 keskeytetään ohjelma. Esimerkkiajossa on koneelta kysytty muuttujan SAT arvoa (PRINT SAT). Sen jälkeen jatketaan CONT-käskyllä komentotilasta ja suoritus siirtyy riville 50, jolla on REM-lause. Rivillä 60 tulostetaan arvottu luku 100-kertaisena. Ohjelma päättyy rivillä 70 END-käskyyn (ei pakollinen).

SWAP

MUOTO: SWAP <muuttuja1>,<muuttuja2>

VERSIOT: MSX: KASET. + LEVYKE
SVI: KASET. + LEVYKE

TARKOITUS: Käskyllä vaihdetaan kahden muuttujan arvot keskenään.

LISÄYKSET: <muuttuja1> = saa arvokseen <muuttuja2>.
<muuttuja2> = saa arvokseen <muuttuja1>.
Molempien muuttujien tulee olla samaa tyyppiä tai tulostuu virheilmoitus "Type mismatch".

ESIM:

```
10  CLS
20  INPUT "ANNA KAKSI LUKUA (PILKKU VÄLIIN) "; A , B
30  PRINT "ANNAN MUUTTUJALLE A ARVOKSI"; A
40  PRINT "JA MUUTTUJALLE B ARVOKSI"; B
50  PRINT
60  PRINT "VAIHDAMME MUUTTUJIEN ARVOT"
70  PRINT "KESKENÄÄN KÄYTTÄEN SWAP-KÄSKYÄ."
80  SWAP A,B
90  PRINT
100 PRINT "MUUTTUJAN A ARVO ON NYT"; A
110 PRINT "JA MUUTTUJAN B ARVO ON"; B
120 PRINT "VAIHTO SIIS ONNISTUI !"
130 END
```

SWITCH

MUOTO: SWITCH/SWITCH STOP

VERSIOT: MSX: ei käytössä
SVI: KASET. + LEVYKE

TARKOITUS: Käskyllä aktivoidaan toinen muistipankki (SWITCH) tai palataan ensimmäiseen muistipankkiin (SWITCH STOP).

LISÄYKSET: SWITCH-käskyllä aktivoidaan seuraava muistipankki. Jos koneeseen ei ole kytketty lisämuistia (64 ktavua), tulostuu virheilmoitus "Illegal function call". Jos lisämuisti on kytketty, tulostuu uuden muistipankin numero.
SWITCH STOP -käskyllä palataan edelliseen muistipankkiin. Jos käytössä on vain yksi lisämuisti, voidaan muistipankista kaksi (2) palata muistipankkiin yksi (1) myös pelkällä SWITCH-komennolla. Siirryttäessä pankista toiseen palauttaa SWITCH-käsky suorituksen komentotilaan.
Jokainen mustipankki voi sisältää rivit 0—65529. Voit säilyttää kahta eri ohjelmaa samoilla rivinumeroilla eri muistipankeissa. Editointi muistipankissa yksi vaikuttaa vain pankin yksi ohjelman sisältöön.
Mikä on muistipankki?
Muistipankilla tarkoitetaan 64 kilotavun (merkitään usein 64 ktav, sisältää tilaa 65536:lle merkille, numerolle tai kirjaimelle) suuruista muistin osaa. Tästä muistimäärästä varaa BASIC oman tilansa.
Mikrotietokoneen kyky käsitellä tietoa kerrallaan on rajallinen ja rajallinen on myös nk. osoiteavaruus, johon tietoa (esim. ohjelmia) voidaan taltioida. SPECTRAVIDEO ja MSX-mikrot käyttävät molemmat Z-80A mikroprosessoria, keskusyksikköä, joka on maailman eniten käytetty 8-bittinen keskusyksikkö. Kahdeksan bittisellä keskusyksiköllä tarkoitetaan mikroprosessoria, joka käsittelee tietoa kahdeksalla nollan ja ykkösen yhdistelmällä. Jokaista merkkiä ja numeroa vastaa oma nollien ja ykkösien koodi.
Kahdeksan bittisen keskusyksikön muistiavaruus, jota kerrallaan voidaan hallita, on kooltaan 65536 muistipaikkaa sisältävä lokerikko (luku pyöristetään usein 64-kilotavuksi). Lokerikon jokainen lokero sisältää tietoa yhden merkin/tavun verran. Lokerikosta käytetään nimitystä muistipankki.

ESIM:

KYTKE MUISTINLAAJENNUS KIINNI.
(joko laajennusyksiköllä SV-602/SV-601/SV-605X.)
HUOM! KYTKE VIRTA POIS KAIKISTA LAITTEISTA LIITTÄESSÄSI LISÄMUISTIA.
KYTKE VIRTA.

Kirjoita (muistipankkiin 1) seuraava ohjelma:

```
10 REM  TÄMÄ ON PANKKI 1
20 PRINT "PANKKI NUMERO 1."
```

Anna käsky:

SWITCH <ENTER>

Nyt pitäisi kuvaruutuun tulostua tunnus muistipankista kaksi (2). Jos ei, tarkista liitännät. Kirjoita muistipankkiin 2 seuraava ohjelma:

```
10 REM  TÄMÄ ON PANKKI NRO 2
20 PRINT "TÄMÄ ON PANKKI 2."
```

Kokeile nyt käskyjen SWITCH ja SWITCH STOP vaikutusta. Käytä LIST-käskyä ja huomaa siirtyminen pankista toiseen. Kokeile myös NEW-käskyä ja huomaa sen vaikutus vain siinä pankissa, jossa käsky annetaan.

TROFF/TRON

MUOTO: **TROFF**
TRON

VERSIOT: **MSX: KASET. + LEVYKE**
SVI: KASET. + LEVYKE

TARKOITUS: Käskyllä (TRON) voidaan seurata ohjelmarivien suoritusta kuvaruudulla. TROFF-käsky eliminoi TRON-käskyn.

LISÄYKSET: Käskyparia käytetään syntaksi- eli kielioppivirheiden etsimiseen ohjelmasta. TRON-käskyn jälkeen tulostuu kuvaruudulle suoritettava rivinumero hakasulkujen sisällä. TROFF-käskyllä kytketään toiminto pois.
(TRON = trace on)
(TROFF = trace off)

ESIM:

```
10 PRINT "RIVI KYMMENEN"
20 PRINT "SILMUKKA . . ."
30 FOR A%=1 TO 2000 : NEXT
40 GOTO 10

TRON <ENTER>
Ok
RUN <ENTER>
[10] RIVI KYMMENEN
[20] SILMUKKA . . .
[30] [50] [10]RIVI KYMMENEN
[20] SILMUKKA . . .
```

Rivit 10 ja 20 tulostavat tekstiä. Rivillä 30 on FOR-NEXT -silmukka, jolla kulutetaan tällä kertaa vain aikaa. Käsky näet antaa kokonaislukumuuttujalle A arvot 1—2000, joka vie oman aikansa. Rivillä 40 palataan takaisin riville 10 GOTO-käskyllä. Kun olet kirjoittanut ohjelman, anna TRON-käsky. Sitten käynnistetään ohjelma RUN-käskyllä. Nyt tulostuu kuvaruudulle hakasulkujen sisällä rivinumero ja mahdollinen PRINT-lause. Kuten huomaat, aikaa kuluu eniten rivillä 30. Ohjelman voi pysäyttää painamalla CTRL+STOP.

ARITMEETTISET FUNKTIOT

Sekä MSX- että SVI-BASIC sisältää kaikki tarvittavat matemaattiset funktiot. Aritmeettiset funktiot suoritetaan käyttäen kaksinkertaista tarkkuutta, paitsi ABS-, FIX- ja INT-funktioissa, joissa tuloksen tarkkuus riippuu argumentin eli laskettavan luvun tarkkuudesta.
CINT-, CDBL- ja CSNG-funktioiden tarkkuus riippuu itse funktiosta.
Kaikilla funktioilla laskettava luku, vakio, muuttuja tai niistä muodostettu lauseke sijoitetaan sulkuihin. Sulkujen määrää ei ole rajoitettu ja sulkuja käytettäessä aloitetaan laskeminen sisimmästä sulkulausekkeesta.
Käytössä olevat funktiot (MSX ja SVI) ovat seuraavat:

ABS ATN BIN$ CDBL CINT COS CSNG EXP FIX FRE
HEX$ INT LOG OCT$ RND SGN SIN SQR TAN TIME

Mukana on myös muuttujan TIME määritys sekä lukujärjestelmämuunnoskäskyt BIN$, HEX$ ja OCT$

ABS

MUOTO: **ABS (X)**

VERSIOT: **MSX: KASET. + LEVYKE**
SVI: KASET. + LEVYKE

TARKOITUS: Funktio palauttaa X:n itseisarvon.

LISÄYKSET: X = luku, muuttuja tai lause, josta itseisarvo palautetaan.

ESIM:

```
10 A=.15
20 PRINT ABS (A), ABS )A* —50)

RUN <ENTER>
 .15          7.5
Ok
```

ATN

MUOTO: ATN (X)

VERSIOT: MSX: KASET. + LEVYKE
SVI: KASET. + LEVYKE

TARKOITUS: Funktio antaa ilmaisun X arkustangentin radiaaneissa.

LISÄYKSET: Palautettava arvo on välillä: (—pii/2 < ATN (X) < pii/2).

ESIM:

```
PRINT ATN (1) <ENTER>
 .78539816339745
Ok
```

BIN $

MUOTO: BIN$ (X)

VERSIOT: MSX: KASET. + LEVYKE
SVI: KASET. + LEVYKE

TARKOITUS: Funktiolla muunnetaan ilmaisun X arvo binääriluvuksi.

LISÄYKSET: Ilmaisun X arvo tulee olla välillä —32768 — +65535.
Jos X on desimaaliluku, jätetään desimaaliosa huomioimatta (pois). Jos taas X on negatiivinen luku, käytetään vastauksessa toisenkomplementtia.
Tällöin esim. BIN$ (—2) = BIN$ (65536—2).
Binaarilukuja tulostettaessa jätetään merkitsemättömät nollat tulostamatta.
Lukujärjestelmien yhtäläisyyksistä.

DESIMAALILUKU	BINAARILUKU
—2	1111 1111 1111 1110
—1	1111 1111 1111 1111
0	0000 0000 0000 0000
1	0000 0000 0000 0001
2	0000 0000 0000 0010
3	0000 0000 0000 0011
10	0000 0000 0000 1010
100	0000 0000 0110 0100

ESIM:

```
PRINT BIN$ (10)  <ENTER>
1010
Ok

PRINT BIN$ (27*—5) <ENTER>
1111111101111001
Ok
```

CDBL

MUOTO: CDBL (X)

VERSIOT: MSX: KASET. + LEVYKE
SVI: KASET. + LEVYKE

TARKOITUS: Funktiolla muunnetaan annettu luku kaksinkertaisen tarkkuuden luvuksi.

LISÄYKSET: Sekä MSX- että SVI-BASIC käyttää oletusarvona kaksinkertaista tarkkuutta.

ESIM:

```
10  A%=1 : B%=7
20  PRINT CDBL (A%/B%)

RUN <ENTER>
 .14285714285714
Ok
```

Rivillä 10 määritellään kokonaislukumuuttujat A ja B. Rivillä 20 halutaan muuttujien osamäärä tulostaa kuitenkin käyttäen kaksinkertaista tarkkuutta.

CINT

MUOTO: CINT (X)

VERSIOT: MSX: KASET. + LEVYKE
SVI: KASET. + LEVYKE

TARKOITUS: Funktio antaa positiivisilla luvuilla suurimman kokonaisluvun, joka on <= X ja negatiivisilla luvuilla suurimman kokonaisluvun, joka on >= X. Vrt. INT-funktioon.

LISÄYKSET: Ilmaisu X voi olla väliltä —32768 — +32767.

ESIM:

```
10 A=3.49 : B=7.99
20 PRINT CINT (A) , CINT (—A)
30 PRINT CINT (B) , CINT (—B)

RUN <ENTER>
 3            —3
 7            —7
Ok
```

Huomaa ero negatiivisilla arvoilla!
CINT-funktiota voidaan käyttää laskutoimitusten nopeuttamiseen, jos vain tuloksen kokonaisosa on tärkeä.

```
100 A%=CINT (B#) + CINT (C#)
```

COS

MUOTO: COS (X)

VERSIOT: MSX: KASET. + LEVYKE
SVI: KASET. + LEVYKE

TARKOITUS: Funktio antaa kulman X kosinin radiaaneissa.

LISÄYKSET:

```
1 radiaani  = 0.0174533    astetta
1   aste    = 57.29578     radiaania
1   pii     = 3.141592654
1 täysi ympyrä = 2 * pii radiaania
```

ESIM:

```
PRINT COS (45) <ENTER>
 .52532198881661
Ok
```

CSNG

MUOTO: CSNG (X)

VERSIOT: MSX: KASET. + LEVYKE
SVI: KASET. + LEVYKE

TARKOITUS: Funktio muuntaa ilmaisun X yksinkertaisentarkkuudenluvuksi.

LISÄYKSET: Yksinkertaisentarkkuudenluvut esitetään käyttäen kuutta (6) merkitsevää numeroa.

ESIM:

```
PRINT CSNG (3.141592653) <ENTER>
 3.14159
Ok
```

EXP

MUOTO: EXP (X)

VERSIOT: MSX: KASET. + LEVYKE
SVI: KASET. + LEVYKE

TARKOITUS: Funktio antaa luonnollisen luvun "e" korotettuna potenssiin X.

LISÄYKSET: Ilmaisun X tulee olla X <= 145.06286085862.

ESIM:

```
PRINT EXP (1) <ENTER>
 2.7182818284588
Ok
```

FIX

MUOTO: FIX (X)

VERSIOT: MSX: KASET. + LEVYKE
SVI: KASET. + LEVYKE

TARKOITUS: Funktio palauttaa ilmaisun X kokonaisosan.

LISÄYKSET: Funktio voidaan merkitä yhtäsuureksi seuraavasti:

FIX (X) = SGN (X) * INT (ABS (X))

ESIM:

```
10 PRINT "FIX- JA INT-FUNKTIOIDEN ERO"
20 PRINT
30 PRINT "INT (—3.14) ="; INT (—3.14)
40 PRINT "FIX (—3.14) ="; FIX (—3.14)
```

Rivillä 10 tulostetaan otsikko. Rivillä 20 tulostetaan yksi tyhjä rivi. Rivit 30 ja 40 tulostavat eron kahden funktion välillä, joka näkyy negatiivisilla luvuilla.

FRE

MUOTO: FRE (<muuttuja>)

VERSIOT: MSX: KASET. + LEVYKE
SVI: KASET. + LEVYKE

TARKOITUS: Funktio antaa vapaan RAM-muistitilan, joka on BASIC-ohjelmien käytettävissä.

LISÄYKSET: FRE (X) -käsky antaa muistitilan, joka on BASIC- ja konekieliohjelmien käytettävissä. Ilmaisu X on jokin luku.
FRE (X$) -käsky antaa muistitilan, joka on merkkijonomuuttujan X käytettävissä. Jos CLEAR-käskyllä ei ole tehty erikseen merkkijonolle X tilaa, on perustilanteessa käytettävissä tilaa 200 merkkiä. CLEAR-käskyllä voidaan jokaista merkkijonoa varten varata 254 merkkiä tilaa.

ESIM:

```
PRINT FRE (0) <ENTER>
 28815
Ok

PRINT FRE (" ") <ENTER>
 200
Ok
```

Jälkimmäinen (PRINT FRE (" "))-lause palauttaa merkkijonoille vapaan muistitilan.

HEX $

MUOTO: HEX$ (X)

VERSIOT: MSX: KASET. + LEVYKE
SVI: KASET. + LEVYKE

TARKOITUS: Funktio muuntaa ilmaisun X heksadesimaalijärjestelmän luvuksi.

LISÄYKSET: Ilmaisun X tulee olla välillä —32768 — +65535. X:n ollessa desimaaliluku, jätetään luvusta desimaaliosa pois muunnosta tehtäessä.
Jos ilmaisu X on negatiivinen luku, ilmoitetaan vastaus toisenkomplementtina. Tällöin HEX$ (—X) = HEX$ (65536—X).
Heksadesimaalijärjestelmä on 16-kantainen lukujärjestelmä, jossa luvut esitetään numeroilla 0—9 ja kirjaimilla A—F.
Lukujärjestelmät (desimaali ja heksa) vastaavat toisiaan seuraavasti:

DES:	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	...
HEK:	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	10	11	12	...

Eri lukujärjestelmien luvut merkitään seuraavasti:

DESIMAALILUKU:	0	10	100	—100
HEKSADESIMAALILUKU:	&H0	&H1A	&HFF00	
BINAARILUKU:	&B0011	&B10101100	&B10	

ESIM:

```
10  FOR A% = —100 TO 100 STEP 10
20  PRINT TAB (10); A% ; TAB (16); HEX$ (A%)
30  NEXT A%
```

Rivillä 10 määritellään FOR-NEXT -silmukka, jossa silmukkalaskuri A saa arvot —100 — +100 kymmenen välein. Rivillä 20 määrätään tulostumaan sarakkeeseen 10 desimaaliluku, joka on silmukkalaskurin arvo ja sarakkeeseen 16 vastaava heksadesimaaliluku. Rivin 30 NEXT-käsky sulkee silmukan.

INT

MUOTO: INT (X)

VERSIOT: MSX: KASET. + LEVYKE
SVI: KASET. + LEVYKE

TARKOITUS: Funktio antaa suurimman kokonaisluvun, joka on <= X.

LISÄYKSET: Funktion avulla voidaan suorittaa pyöristys ylöspäin luvusta X:

```
Y= —INT (—X)
```

Saatu arvo palautetaan muuttujalle Y.
Funktiolla saadaan pyöristys tietyllä tarkkuudella. Esimerkki kahdella desimaalilla suoritettavasta pyöristyksestä:

```
Y= .01*INT (100*X+.5)
```

ESIM:

```
10  INPUT " ANNA PYÖRISTETTÄVÄ LUKU "; A
20  PRINT
30  INPUT "ANNA TARKKUUS, JOLLA PYÖRISTETÄÄN "; B
40  C=10potB
50  D=INT (C*A+.5)/C
60  PRINT
70  PRINT "VASTAUS:"; D
```

Tämä esimerkkiohjelma pyöristää haluamasi luvun haluamallasi tarkkuudella. Rivillä 10 syötetään INPUT-käskyllä pyöristettävä luku. Rivillä 20 tulostetaan tyhjä rivi. Rivillä 30 syötetään tarkkuus, jolla pyöristys halutaan suorittaa. Riveillä 40 ja 50 suoritetaan pyöristys edellä esitetyn kaavan mukaisesti. Rivin 60 PRINT-käsky tulostaa tyhjän rivin. Rivillä 70 tulostetaan vastaus.

LOG

MUOTO: LOG (X)

VERSIOT: MSX: KASET. + LEVYKE
SVI: KASET. + LEVYKE

TARKOITUS: Funktio palauttaa ilmaisun X luonnollisen ("e"-kantaisen) logaritmin.

LISÄYKSET: Ilmaisun X tulee olla > 0.
Tarvittaessa muita, kuin e-kantaisia logaritmeja, jaetaan ilmaisun X luonnollinen logaritmi halutun kantaluvun luonnollisella logaritmilla.
LOG- ja EXP-funktiot ovat toistensa käänteisfunktioita.
Lauseella LOG (EXP (X)) saadaan luku X.
10-kantainen logaritmi luvusta X saadaan lauseella:

```
LOG (X) / LOG (10)
```

ESIM:

```
PRINT LOG (1) <ENTER>
 0
Ok

PRINT LOG (10) <ENTER>
 2.302585092994
Ok
```

OCT $

MUOTO: OCT$ (X)

VERSIOT: MSX: KASET. + LEVYKE
SVI: KASET. + LEVYKE

TARKOITUS: Funktio muuntaa ilmaisun X oktaalijärjestelmän luvuksi.

LISÄYKSET: Ilmaisun X tulee olla välillä —32768 — +65535.
Jos X on negatiivinen luku, käytetään vastauksessa toisenkomplementtia. Tällöin OCT$ (—X) = OCT$ (65536—X).
Eri lukujärjestelmien luvut esitetään seuraavasti:

DESIMAALIJÄRJESTELMÄ: X% ; kokonaisluku.
X! ; yksinkertaisentarkkuuden luku.
X# ; kaksinkertaisentarkkuuden luku.
(tai X)

BINAARIJÄRJESTELMÄ: luvut &B0000000000000000 — &B1111111111111111

HEKSADESIMAALIJÄRJESTELMÄ: luvut &H0000 — &HFFFF

OKTAALIJÄRJESTELMÄ: luvut &O000000 — &O177777

Desimaali- ja oktaaliluvut vastaavat toisiaan seuraavasti:

DES:	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**. . .**
OKT:	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**20**	**21**	**. . .**

ESIM:

```
PRINT OCT$ (100) <ENTER>
144
Ok

PRINT OCT$ (&B1010) <ENTER>
12
Ok

PRINT OCT$ (&HA0FF) <ENTER>
120377
Ok
```

Huomaa kaksi viimeistä esimerkkiä, joissa muunnetaan ensin binaariluku ja jälkimmäisessä heksaluku oktaaliluvuksi.

```
10  FOR A% = 1 TO 5
20  READ B$
30  PRINT VAL ("&O" + B$);
40  NEXT
50  DATA 23,56,77,70,0
RUN <ENTER>
 19  46  63  56  0
Ok
```

Tämä ohjelma lukee DATA-lauseilta viisi (5) oktaalilukua, jotka tulostetaan desimaalilukuina. Rivit 10—40 muodostavat FOR-NEXT -silmukan. Rivillä 10 määritellään laskurille A arvot yhdestä viiteen. Rivillä 20 luetaan READ-käskyllä yksi DATA riviltä 50. DATA on merkitty merkkijonoksi, jotta seuraavalla rivillä oleva tyyppimuunnos olisi mahdollinen. Rivillä 30 suoritetaan luetulle merkkijonolle (joka tällä kertaa on luku) muunnos seuraavasti. Luettuun DATAan lisätään oktaaliluku nolla (&O= 0) ja näin itseasiassa kerrotaan tietokoneelle luetun luvun olevan oktaaliluku. Rivin 40 NEXT-käsky sulkee silmukan. Rivin 50 DATAt on erotettuina pilkulla.

RND

MUOTO: RND (X)

VERSIOT: MSX: KASET. + LEVYKE
SVI: KASET. + LEVYKE

TARKOITUS: Funktio generoi (tuottaa) satunnaisluvun väliltä 0 <RND<1.

LISÄYKSET: Jos X>0, annetaan uusi satunnaisluku.
Jos X=0, toistuu edellinen satunnaisluku.
Jos X<0, antaa BASIC satunnaislukugeneraattorille uuden kantaluvun. Koska RND-funktiolla saatavat luvut ovat joka ajokerralla samanlaisia, on haluttaessa välttää samojen satunnaislukujen toisto annettava ennen uutta ajoa generaattorille uusi kantaluku (negatiivinen). Myös kantalukua on vaihdettava uusittaessa ajoa, joten samaa negatiivista kantalukua voidaan käyttää vain kerran. Usein käytetään uuden kantaluvun tuottamiseen käskyä:

RND (-TIME)

TIME on BASIC:n oma kokonaislukulaskuri, joka kasvaa yhdellä 50-kertaa sekunnissa. Näin ollen on erittäin epätodennäköistä, että TIME:n arvo olisi kahdella ajokerralla sama.

ESIM:

```
10 FOR A%=1 TO 3
20 PRINT RND (1)
30 NEXT
40 PRINT
50 PRINT RND (0)
60 PRINT
70 PRINT RND (-1)
80 PRINT RND (-1)
90 PRINT RND (-2)

RUN <ENTER>
 .32145547165429
 .05478115527
 .9012657666566

 .9012657666566

 .40032885670122
 .40032885670122
 .11297823255011
Ok
```

Rivit 10—30 muodostavat FOR-NEXT -silmukan, joka arpoo kolme satunnaislukua ja tulostaa ne (rivi 20) allekkain. Rivit 40 ja 60 tulostavat tyhjän rivin. Rivillä 50 arvotaan (toistetaan) edellinen satunnaisluku RND (0) -funktiolla. Rivillä 70 annetaan generaattorille uusi kantaluku, mutta koska rivillä 80 annetaan sama kantaluku uudestaan, tulostuu sama satunnaisluku. Rivi 90 antaa uuden kantaluvun.

```
10 FOR A%=1 TO 7
20 PRINT TAB (A%*4); INT (37*RND (A%)+.5);
30 NEXT
40 PRINT : PRINT
50 A=RND (-TIME)
60 GOTO 10
```

Esimerkki yksinkertaisesta lotto-ohjelmasta. Ohjelmassa on parantamisen varaa, sillä se saattaa tuottaa numeron nolla (0) tai mahdollisesti yhteen riviin kaksi samaa numeroa. Rivit 10—30 muodostavat FOR-NEXT -silmukan, jonka aikana lottonumerot tuotetaan. Rivillä 10 määritellään riviin kuuluvan seitsemän (7) numeroa. Rivillä 20 tulostetaan käyttäen TAB-käskyä (sarakkeeseen muuttujan A arvo kertaa neljä) numero, joka saadaan INT-lauseesta. Koska lottonumeron tulee olla välillä 1—37, on satunnaisluku kerrottava luvulla 37 ja tähän lisättävä luku puoli (.5), johtuen INT-funktion suorittamasta pyöristyksestä. Rivin 30 NEXT-käsky sulkee silmukan. Rivillä 40 tulostetaan kaksi (2) tyhjää riviä. Rivillä 50 annetaan generaattorille uusi kantaluku jokaista uutta riviä varten. Rivillä 60 annetaan käsky siirtyä ohjelman alkuun. Ohjelman saat päättymään painamalla CTRL+STOP -painikkeita (Break).

SGN

MUOTO: SGN (X)

VERSIOT: MSX: KASET. + LEVYKE
SVI: KASET. + LEVYKE

TARKOITUS: Funktio palauttaa ilmaisun X etumerkin.

LISÄYKSET: Jos X>0, on SGN (X) = 1.
Jos X=0, on SGN (X) = 0.
Jos X<0, on SGN (X) = —1.

ESIM:

```
10  INPUT "ANNA JOKIN LUKU "; A
20  ON SGN (A)+2 GOTO 30,50,70
30  PRINT "ANTAMASI LUKU ON NEGATIIVINEN."
40  END
50  PRINT "ANTAMASI LUKU ON NOLLA."
60  END
70  PRINT "ANTAMASI LUKU ON POSITIIVINEN."
80  END
```

Rivillä 10 pyydetään käyttäjää syöttämään jotain lukua. Luvun ja ENTER:n painamisen jälkeen siirrytään riville 20, jolla ohjelma haarautuu kolmeen vaihtoehtoon, riippuen annetusta luvusta. Jos annetaan esim. jokin negatiivinen luku, antaa funktio SGN (neg. luku) arvon —1 ja kun tähän lisätään luku 2, saadaan vastaava rivi (eli luettelon ensimmäinen —1+2=1), jolle suoritus siirtyy GOTO-käskyllä. Riville, jolle siirrytään, tulostuu teksti annetun luvun mukaan ja seuraavalla rivillä ohjelma päättyy END-käskyyn.

SIN

MUOTO: SIN (X)

VERSIOT: MSX: KASET. + LEVYKE
SVI: KASET. + LEVYKE

TARKOITUS: Funktio antaa kulman X sinin radiaaneissa.

LISÄYKSET: Palautettava arvo on välillä —pii/2 < SIN (X) < pii/2.

ESIM:

```
PRINT SIN(45) <ENTER>
 .85090352453482
Ok
```

SQR

MUOTO: SQR (X)

VERSIOT: MSX: KASET. + LEVYKE
SVI: KASET. + LEVYKE

TARKOITUS: Funktio antaa ilmaisun X neliöjuuren.

LISÄYKSET: ILMAISUN X TULE OLLA X>=0.
SQR-funktion voi korvata Xpot0.5 -lauseella, mutta SQR-funktio on noin 40 prosenttia nopeampi tapa.

ESIM:

```
PRINT SQR (7) <ENTER>
 2.6457513110646
Ok
```

TAN

MUOTO: TAN (X)

VERSIOT: MSX: KASET. + LEVYKE
SVI: KASET. + LEVYKE

TARKOITUS: Funktio antaa kulman X tangentin radiaaneissa.

LISÄYKSET: TAN (X) = SIN (X) / COS (X)

ESIM:

```
PRINT TAN (45) <ENTER>
 1.6197751905486
Ok
```

TIME

MUOTO: TIME

VERSIOT: MSX: KASET. + LEVYKE
SVI: KASET. + LEVYKE

TARKOITUS: Koneen sisäinen kello/laskuri.

LISÄYKSET: TIME-funktio ei ole varsinainen kello, mutta sen avulla voidaan toteuttaa tarkka kello tai ajastin.
Funktio palauttaa kokonaisluvun, joka kasvaa yhdellä 50 kertaa sekunnissa. Saavuttaessaan arvon 65535, alkaa laskenta alusta eli nollasta (0).
TIME-laskurin päivitys (kasvattaminen) keskeytyy mm. nauhalta latauksen/tallennuksen yhteydessä.
TIME-laskurille voidaan syöttää mikä tahansa kokonaisluku väliltä 0—65535.

ESIM:

```
10  TIME=0
20  LOCATE 18,11
30  PRINT TIME
40  GOTO 20
```

Rivillä 10 nollataan TIME-laskuri. Rivillä 20 kohdistetaan tulostus riville 11, sarakkeeseen 18. Seuraavalla rivillä annetaan käsky tulostaa TIME-laskurin sen hetkinen arvo. Rivillä 40 hypätään takaisin riville 20 ja kohdistetaan tulostus jne. Ohjelma pysähtyy painamalla CTRL+STOP -painikkeita.

MERKKIJONOFUNKTIOT

ASC

MUOTO: ASC (<string>)

VERSIOT: MSX: KASET. + LEVYKE
SVI: KASET. + LEVYKE

TARKOITUS: Funktio palauttaa merkkijonon ensimmäisen merkin ASCII-koodin (kts. ASCII-kooditaulukko).

LISÄYKSET: <string> = merkkijono, jonka ensimmäisestä merkistä koodiluku palautetaan.
Palautettava ASCII-koodi on kymmenjärjestelmän luku. Sen muuntaminen heksadesimaaliluvuksi voidaan suorittaa HEX$-funktiolla.

ESIM:

```
10  A$="ESIMERKKI"
20  PRINT "MERKKIJONON ESIMERKKI"
30  PRINT "ENSIMMÄISEN MERKIN ELI E:n"
40  PRINT "ASCII-KOODI ON"; ASC (A$)
50  PRINT "JA HEKSANA"; HEX$ (ASC (A$))

RUN <ENTER>
MERKKIJONON ESIMERKKI
ENSIMMÄISEN MERKIN ELI E:n
ASCII-KOODI ON 69
JA HEKSANA 45
Ok
```

Rivillä 10 määritellään merkkijono A:ksi sana "ESIMERKKI". Rivit 20 ja 30 tulostavat tekstiä. Rivillä 40 tulostetaan ensin tekstiä ja sitten "E":n ASCII-koodi. Rivillä 50 vastaavan kirjaimen heksakoodi. Huom! erot pienten ja isojen kirjainten välillä.
Käyttäen apuna RIGHT$-, LEFT$- ja MID$-funktioita saadaan merkkijonosta poimittua minkä tahansa merkin ASCII-koodi.

```
10  INPUT "ANNA ETUNIMESI :"; NIMI$
20  PRINT "ETUNIMESI TOISEN KIRJAIMEN"
30  PRINT "ASCII-KOODI ON"; ASC (MID$ (NIMI$,2))

RUN <ENTER>
ANNA ETUNIMESI :? Heikki <ENTER>
ETUNIMESI TOISEN KIRJAIMEN
ASCII-KOODI ON 101
Ok
```

Rivillä 10 pyydetään käyttäjää syöttämään jokin nimi. Rivillä 20 tulostuu tekstiä. Rivillä 30 tulostetaan tekstin jälkeen syötetyn merkkijonon toisen merkin ASCII-koodi. Seuraava esimerkki muuntaa syöttämäsi merkkijonon muotoon, jota voisi verrata salakoodiin.

```
10  CLEAR 1000
20  LINE INPUT "ANNA KOODATTAVA TEKSTI : "; A$
30  PRINT
40  FOR B%=1 TO LEN (A$)
50  C$=MID$ (A$,B%,1)
60  D%=ASC (C$) + 47
70  IF D%>255 THEN D%=D%—256
80  E$=E$+CHR$ (D%)
90  NEXT
100 PRINT "JA TÄSSÄ SYÖTTÄMÄSI"
110 PRINT "TEKSTI KOODATTUNA :"
120 PRINT
130 PRINT E$
```

Rivillä 10 varataan tarvittaville muuttujille tilaa CLEAR-käskyllä. Rivillä 20 syötetään koodattava teksti LINE INPUT -lauseella. Rivillä 30 tulostetaan yksi tyhjä rivi. Rivit 40—90 muodostavat FOR-NEXT -silmukan, jonka aikana syötetty teksti koodataan. Syötetyn merkkijonon pituus saadaan LEN-funktiolla (vrt. rivi 40). Koodaus suoritetaan siten, että syötetyn merkkijonon jokaisen merkin ASCII-koodiin lisätään koodiluku 47 (vrt. rivi 60), joka voi olla mikä tahansa koodiluku tai mahdollisesti jonkin säännön mukaisesti muuttuva koodiluku, jolloin koodin purkaminen on kielletyille henkilöille vieläkin vaikeampaa. Rivillä 70 varmistetaan, ettei saatu ASCII-koodi ylitä 255, sillä se ASCII-kooditaulukon suurin mahdollinen luku. Rivillä 80 talletetaan saatu CHR$-koodi muuttujaan E$. Rivin 90 NEXT-käsky sulkee silmukan. Rivit 100—120 tulostavat otsikon ja rivi 130 koodatun tekstin.
Jos haluat dekoodata tekstin, kirjoita seuraava ohjelma:

```
140 PRINT "JOS HALUAT PURKAA KOODAUKSEN,"
150 PRINT " PAINA JOTAIN NÄPPÄINTÄ."
160 A$=INKEY$ : IF A$="" THEN 160
170 FOR B%=1 TO LEN (E$)
180 C$=MID$ (E$, B%,1)
190 D%=ASC (C$) — 47
200 IF D%<0 THEN D%=D%+256
210 A$=A$+CHR$ (D%)
220 NEXT
230 PRINT
240 PRINT "TÄSSÄ ALKUPERÄINEN TEKSTI :"
250 PRINT
260 PRINT A$
```

Ohjelma muuten samoin, kuin edellinen, mutta nyt koodiluku 47 vähennetään jokaisen merkin ASCII-koodista. Rivi 160 muodostaa silmukan, joka odottaa, että jotain näppäintä painetaan.

CHR $

MUOTO: CHR$ (X)

VERSIOT: MSX: KASET. + LEVYKE
SVI: KASET. + LEVYKE

TARKOITUS: Funktio palauttaa ASCII-koodia X vastaavan merkin.

LISÄYKSET: Ilmaisun X tulee olla 0 <= X <=255 tai tulostuu virheilmoitus "Illegal function call". Merkit 0—31 eivät tulostu kuvaruutuun näkyvinä merkkeinä (paitsi ohjainkoodit kursoria varten). Näitä koodeja käytetään mm. ohjattaessa kirjoitinta tai tekstinkäsittelyohjelmissa siirtämään kursoria.
CHR$-funktiota käytetään merkin lähettämiseen I/O-laitteelle, kuten näyttöpääte, kirjotin, jne. Merkki lähetetään PRINT-käskyn avulla.
Tulostettaessa koodinumeroita 128—255, eroaa SVI BASIC ja MSX BASIC toisistaan johtuen eri grafiikkamerkeistä.
CHR$-funktion käänteisfunktio on ABC.

ESIM:

```
PRINT CHR$ (69)+CHR$ (46)+CHR$ (84)+CHR$ (46) <ENTER>
E.T.
Ok
```

CHR$-funktioita voidaan siis yhdistää, kuten mitä tahansa PRINT-lauseen ilmaisuja, käyttäen merkkejä: + ; , jne.

INKEY $

MUOTO: X$=INKEY$

VERSIOT: MSX: KASET. + LEVYKE
SVI: KASET. + LEVYKE

TARKOITUS: Funktio palauttaa yhden merkin merkkijonolle X, jota käskyn suorituksen aikana on painettu.

LISÄYKSET: Jos käskyn suorituksen aikana ei ole painettu mitään näppäintä, palautuu merkkijonolle X arvo nolla (NULL).
INKEY$-funktiota käytetään, kun ohjelmassa halutaan syöttää tietoa ilman, että ohjelman suoritus pysähtyy (esim. peleissä, joissa tulee jotain näppäintä painaa oikealla hetkellä).

ESIM:

```
10   PRINT "ANNA TUNNUS"
20   FOR A%=1 TO 6
30   A$=INKEY$ : IF A$="" THEN 30
40   KOODI$=KOODI$+A$
50   NEXT
60   IF KOODI$="OIKEA" OR KOODI$="oikea" THEN 100
70   X=X+1
80   PRINT "VÄÄRÄ TUNNUS !" : PRINT "ANNA UUSI."
90   IF X=3 THEN END ELSE GOTO 10
100  REM  ohjelma alkaa tästä
          ....
```

Liittämällä tämä esimerkkiohjelma ohjelman alkuun voidaan pääsy ohjelmaan estää niiltä, jotka eivät tunne tunnussanaa. Rivillä 10 tulostetaan teksti, jossa pyydetään käyttäjää syöttämään tunnus. Rivit 20—50 muodostavat ohjelmasilmukan, jonka aikana tunnus syötetään. Tunnuksessa on kuusi (6) kirjainta (rivi 20 FOR . . . 1 TO 6), jotka syötetään yksi kerrallaan rivin 30 suorituksen aikana. Rivin 30 lauseet (INKEY$ ja IF . . .) muodostavat yhdessä toiminnon, joka tutkii painetaanko näppäimistöä. Jos näppäimistöä painetaan, saa A$ arvon, joka on eri, kuin tyhjä ("") ja suoritus siirtyy seuraavalle riville. Jos mitään näppäintä ei ole painettu, on IF-lause tosi ja hypätään uudestaan rivin 30 alkuun. Rivillä 40 lisätään muuttujaan KOODI$ painettu merkki (+A$). Rivin 50 NEXT-käsky sulkee silmukan. Rivillä 60 verrataan saatua tunnusta (joka on "kerätty" muuttujaan KOODI) oikeaan tunnukseen (HUOM! vertaus on tehty kahdesti, sillä tunnus saatetaan syöttää myös pienillä kirjaimilla) ja jos syötetty tunnus täsmää, siirtyy suoritus riville 100, jolta alkaa varsinainen ohjelma. Jos syötetty tunnus oli väärä, siirrytään riville 70 ja kasvatetaan laskuria, joka laskee yrityskertoja. Rivillä 80 tulostuu teksti väärästä tunnuksesta ja rivillä 90 tutkitaan, onko yritys jo kolmas. Jos on, päättyy ohjelma END-käskyyn ja jos ei, annetaan mahdollisuus uuteen yritykseen siirtymällä riville 10. Varsinainen ohjelma alkaa riviltä 100.
Seuraavaksi esimerkki ohjelmasta, joka lukee näppäimistöltä merkkejä, kunnes painetaan 'ESC'-näppäintä.

```
10  CLS
20  LOCATE 0,0
30  A$=INKEY$
40  IF A$="" THEN 30
50  IF A$=CHR$(27) THEN 80
60  PRINT A$;
70  GOTO 30
80  END
```

Tätä ohjelmaa voidaan pitää alkeellisena tekstinkäsittelyohjelmana. Mm. tällä periaatteella voit rakentaa tekstinkäsittelyohjelman, joskin tästä puuttuu vielä kaikki. Rivillä 10 tyhjätään näyttö ja rivillä 20 asetetaan kursori näyttöruuduun vasempaan yläkulmaan. Rivi 30 lukee näppäimistön ja rivin 40 IF-lause, onko jotain näppäintä painettu. Jos ei ole, hypätään lukemaan näppäimistöä uudestaan ja on painettu, tutkitaan rivillä 50, onko painettu ESC:iä (ESC:n ASCII-koodi on 27). Jos on, siirrytään riville 80, jolla ohjelma päättyy.

Jos jotain näppäintä on painettu, tulostetaan se rivillä 60 olevalla PRINT-käskyllä (HUOM! puolipiste, jolla saadaan seuraava merkki tulostumaan heti edellisen jälkeen). Rivillä 70 hypätään lukemaan uutta merkkiä.
INKEY$-funktion ominaisuudet ovat siis seuraavat:
* ohjelman suoritus EI pysähdy.
* painettua näppäintä vastaava merkki EI tulostu näytölle ilman lisäkäskyä.
* ENTER:iä EI tarvitse painaa (vrt. INPUT-käskyyn).
Funktion suoritus voidaan keskeyttää painamalla CTRL+STOP.

INPUT $

MUOTO: **INPUT$ (<määrä>)**

VERSIOT: **MSX: KASET. + LEVYKE**
SVI: KASET. + LEVYKE

TARKOITUS: Funktio lukee näppäimistöltä <määrä> kappaletta merkkejä.

LISÄYKSET: <määrä> = syötettävien merkkien määrä.
Funktio voidaan keskeyttää painamalla CTRL+STOP näppäimiä.
Luettavia merkkejä EI kaiuteta näytölle, eikä viimeisen merkin syötön jälkeen tarvitse painaa ENTER:iä

ESIM:

```
10  PRINT "ANNA TUNNUS !"
20  KOODI$=INPUT$ (6)
30  IF KOODI$="TUNNUS" OR KOODI$="tunnus" THEN 60 ELSE X=X+1
40  PRINT "VÄÄRÄ TUNNUS !" : PRINT "ANNA UUSI."
50  IF X=3 THEN END ELSE GOTO 10
60  REM  ohjelma alkaa tästä
          .....
```

Ohjelma pyytää käyttäjää syöttämään tunnusta (rivi 20), jota verrataan oikeaan rivillä 30 (ohjelma on toteutettu aikaisemmin INKEY$-funktiolla). Jos tunnusta ei kolmen yrityksen jälkeen anneta, pysähtyy ohjelma END-käskyyn ja näin pääsy varsinaiseen ohjelmaan estyy.

INSTR

MUOTO: **INSTR ([<X>,] <string1>,<string2>)**

VERSIOT: **MSX: KASET. + LEVYKE**
SVI: KASET. + LEVYKE

TARKOITUS: Funktiolla etsitään merkkijonon <string2> ensimmäinen esiintymiskohta merkkijonossa <string1>.

LISÄYKSET: <X> = merkkijonon <string1> X:s merkki, josta etsiminen aloitetaan. Ilmaisun tulee olla välillä 0 < X <=255.
Ilmaisut <string1> ja <string2> voivat olla merkkijonomuuttujia, -vakioita tai -ilmaisuja. Mikäli merkkijonoa <string2> ei löydy, palautetaan nolla. Näin käy myös silloin, kun X > <string1> tai jos <string1> sekä <string2> ovat tyhjiä tai jos LEN$ (<string1>) = 0. Jos vain ilmaisu <string2> on tyhjä (NULL), palautetaan ilmaisu <X>.

ESIM:

```
10  A$="SPECTRAVIDEO"
20  PRINT INSTR (A$,"RAVI")
30  PRINT INSTR (7,A$,"RAVI")

RUN <ENTER>
 6
 0
Ok
```

Rivillä 10 määritellään merkkijono A. Rivillä 20 etsitään A$:sta kirjainyhdistelmää "RAVI". Rivillä 30 suoritetaan sama etsintä, mutta alkaen merkistä seitsemän (7). Koska "RAVI" löytyy kuudennesta merkistä alkaen, tulostuu esimerkissä ensin 6 (vrt. rivi 20) ja sitten 0, koska vastaavaa yhdistelmää ei löydy seitsemännen merkin jälkeen.

```
10  CLEAR 510
20  PRINT "ANNA MERKKIJONO, JOSTA"
30  PRINT "ETSIMME TOISTA MERKKIJONOA."
40  PRINT
50  INPUT A$
```

```
60  PRINT
70  PRINT "ANNA SANA, JOTA ETSIT"
80  PRINT "YLEMMÄSTÄ MERKKIJONOSTA."
90  PRINT
100 INPUT B$
110 PRINT
120 C%=INSTR(A$,B$)
130 IF C%=0 THEN PRINT "EI LÖYDY." : END
140 PRINT "MERKKIJONO "; B$ ; " ALKAA";
150 PRINT "MERKKIJONON "; A$ ; "MERKISTÄ #";C%
```

Rivit 40,60,90 ja 110 tulostavat tyhjän rivin. Rivillä 10 tehdään tilaa kahdelle merkkijonolle, jotta molemmat voisivat sisältää maksimin 254 merkkiä. Rivit 20,30,70 ja 80 tulostavat ohjeita. Rivillä 50 syötetään merkkijono, josta etsitään rivillä 100 syötettävää merkkijonoa. Rivillä 120 saa muuttuja C tiedon siitä, monennestako merkistä B$ merkkijonossa A$ alkaa. Rivillä 130 tutkitaan, löytyikö B$ ja jos ei löytynyt, tulostetaan se. Rivit 140 ja 150 tulostavat mistä merkistä B$ alkoi.

LEFT $

MUOTO: LEFT$ (<string>,X)

VERSIOT: MSX: KASET. + LEVYKE
SVI: KASET. + LEVYKE

TARKOITUS: Funktio palauttaa merkkijonosta <string> alkaen vasemmalta X kappaletta merkkejä.

LISÄYKSET: Ilmaisun X tulee olla välillä 0 <= X <=255.
Jos X > LEN$ (<string>), palautetaan koko merkkijono. Jos X = 0, palautetaan tyhjä (NULL).

ESIM:

```
10  A$="ESIMERKKI LEFT$-FUNKTIOSTA."
20  PRINT "TULOSTAMME 12 ENSIMMÄISTÄ MERKKIÄ :"
30  PRINT
40  PRINT LEFT$ (A$,12)
```

```
RUN <ENTER>
TULOSTAMME 12 ENSIMMÄISTÄ MERKKIÄ :
ESIMERKKI LE
Ok
```

LEN

MUOTO: LEN (<string>)

VERSIOT: MSX: KASET. + LEVYKE
SVI: KASET. + LEVYKE

TARKOITUS: Funktio palauttaa merkkijonon <string> pituuden.

LISÄYKSET: Pituuteen lasketaan myös tyhjät (välilyönti) ja erikoismerkit, jotka eivät normaalisti tulostu (esim. CHR$-funktiot).

ESIM:

```
10  LINE INPUT "ANNA NIMESI :"; A$
20  PRINT
30  PRINT "NIMESSÄSI ON"; LEN (A$) ;"KIRJAINTA."
```

MID $

MUOTO: MID$ (<string>,X[,Y])

VERSIOT: MSX: KASET. + LEVYKE
SVI: KASET. + LEVYKE

TARKOITUS: Funktio palauttaa alkaen halutusta merkistä halutun määrän merkkejä.

LISÄYKSET: <string> = merkkijono tai merkkijonomuuttuja, josta merkit palautetaan.
X = X:s merkki, josta alkaen merkkejä palautetaan. Tulee olla välillä 1<= X <=255.
Y = merkkien määrä, joka palautetaan. Tulee olla välillä 0 <= Y <=255.
Jos X > merkkijono on <string> pituus, palautetaan nolla (NULL).
Jos ilmaisu Y jätetään antamatta tai jos merkkijonossa <string> alkaen merkistä X on vähemmän merkkejä, kuin ilmaisu Y on, palautetaan kaikki jäljellä olevat merkit.

ESIM:

```
10  A$="SAIPPUAKAUPPIAS"
20  PRINT MID$ (A$,8,8)
RUN <ENTER>
KAUPPIAS
Ok
```

RIGHT $

MUOTO: RIGHT$ (<string>,X)

VERSIOT: MSX: KASET. + LEVYKE
SVI: KASET. + LEVYKE

TARKOITUS: Funktio palauttaa merkkijonosta <string>alkaen oikealta X kappaletta merkkejä.

LISÄYKSET: Ilmaisun X tulee olla välillä 0 <= X <=255.
Jos X = 0, palautetaan tyhjä (NULL).
Jos X = merkkijonon <string> pituus, palautetaan koko merkkijono.

ESIM:

```
10  A$="SPECTRAVIDEO"
20  PRINT RIGHT$ (A$,5)
RUN <ENTER>
VIDEO
Ok
```

SPACE $

MUOTO: SPACE$ (X)

VERSIOT: MSX: KASET. + LEVYKE
SVI: KASET. + LEVYKE

TARKOITUS: Funktiolla tulostetaan X kappaletta tyhjiä.

LISÄYKSET: Ilmaisun X tulee olla välillä 0—255 ja se voi olla vakio tai numeerinen muuttuja/lause. Vrt. SPC-funktioon.

ESIM:

```
10  A$=SPACE$ (7)
20  PRINT A$ ; "MSX BASIC" ; A$ ; "SVI BASIC"
30  PRINT "1234567          1234567"
RUN <ENTER>
              MSX BASIC    SVI BASIC
1234567         1234567
Ok
```

SPC

MUOTO: SPC (X)

VERSIOT: MSX: KASET. + LEVYKE
SVI: KASET. + LEVYKE

TARKOITUS: Funktiolla tulostetaan X kappaletta tyhjiä.

LISÄYKSET: Ilmauksen X tulee olla välillä 0—255.
SPC-funktiota voidaan käyttää vain PRINT- ja LPRINT-lauseiden yhteydessä. Funktio EI tarvitse puolipistettä (;) erottamaan sitä muista tulostettavista lauseista.

ESIM:

```
10  A$="BASIC" : B$="PASCAL"
20  PRINT A$ SPC (3) B$ SPC (3) "COBOL"
RUN <ENTER>
BASIC    PASCAL    COBOL
Ok
```

STR $

MUOTO: STR$ (X)

VERSIOT: MSX: KASET. + LEVYKE
SVI: KASET. + LEVYKE

TARKOITUS: Funktio muuntaa ilmaisun X arvon merkkijonoksi.

LISÄYKSET: Ilmaisu X voi numeerinen vakio tai muuttuja.
Jos X on positiivinen luku, lisätään muunnettavan merkkijonon eteen yksi tyhjä (välilyönti).
STR$-funktion käänteisfunktio on VAL.

ESIM:

```
10  A$=STR$ (7*7)
20  PRINT "7*7=" ; A$
30  PRINT "TEKSTI" ; A$ ; "TEKSTI"

RUN <ENTER>
7*7= 49
TEKSTI 49TEKSTI
Ok
```

Rivin 30 tulostuksesta voi havaita, kuinka positiivista lukua edeltää yksi välilyönti (mutta ei seuraa yhtään).

STRING $

MUOTO: STRING$ (X,Y)
STRING$ (X,<string>)

VERSIOT: MSX: KASET. + LEVYKE
SVI: KASET. + LEVYKE

TARKOITUS: STRING$ (X,Y) -funktio tulostaa X kappaletta merkkejä, joiden ASCII-koodi on Y.
STRING$ (X,<string>) -funktio tulostaa X kappaletta merkkijonon <string> ensimmäistä merkkiä.

LISÄYKSET: Ilmaisujen X ja Y tulee olla välillä 0—255.
Ilmaisu <string> voi olla merkkijonovakio tai -muuttuja.

ESIM:

```
10  A$=STRING$ (7,"*")
20  B$=" REM "
30  PRINT A$ ; A$ ; "*"
40  PRINT STRING$ (5,42) ; B$ ; STRING$ (5,42)
50  PRINT A$ ; A$ ; "*"

RUN <ENTER>
******************
*****  REM  *****
******************
Ok
```

Rivillä 10 määritellään merkkijonomuuttujan A sisällöksi seitsemän (7) tähteä (*). Rivillä 20 määritellään merkkijonomuuttuja B. Rivillä 30 tulostetaan ensin kaksi kertaa peräkkäin muuttuja A$ ja lisäksi vielä yksi tähti eli yhteensä 15 tähteä. Rivillä 40 tulostuu ensin viisi tähteä (HUOM! tähden ASCII-koodi on 42), sitten muuttuja B$ ja uudelleen viisi tähteä. Rivi 50 tulostaa saman, kuin rivi 30.

TAB

MUOTO: TAB (X)

VERSIOT: MSX: KASET. + LEVYKE
SVI: KASET. + LEVYKE

TARKOITUS: Funktion avulla kohdistetaan tulostus haluttuun kohtaan rivillä.

LISÄYKSET: Ilmaisun X tulee olla välillä 0—255.
Funktiota voidaan käyttää vain PRINT- ja LPRINT-käskyjen kanssa.

Jos kursori (kohdistin) on tulostettaessa PRINT-lausetta ohittanut kohdan X, ei TAB-funktiolla ole vaikutusta, joten tulostettava lause tulostuu heti edellisen perään.
Käytettäessä LPRINT-käskyä ohjaa TAB-funktio kirjoittimen kirjoituspäätä.
Tulostus alkaa aina seuraavasta sarakkeesta, jota edelliseen TAB-funktiolla on osoitettu.

ESIM:

```
10  PRINT
20  PRINT "BASIC" ; TAB (20) ; "PASCAL" ; TAB (10) ; "COBOL"
30  PRINT "12345678901234567890"

RUN <ENTER>

BASIC           PASCALCOBOL
12345678901234567890
Ok
```

Huomaa, kuinka rivillä 20 oleva TAB (10) -funktio ei toteudu, vaan sana "COBOL" tulostuu heti edellisen perään. Huomaa myös se, että TAB (20) -funktio aloittaa tulostuksen sarakkeesta 21, eikä sarakkeesta 20.

VAL

MUOTO: VAL (<string>)

VERSIOT: MSX: KASET. + LEVYKE
SVI: KASET. + LEVYKE

TARKOITUS: Funktio palauttaa merkkijonon <string> sisältämän numeerisen arvon.

LISÄYKSET: Jos merkkijonon <string> ensimmäinen merkki EI ole numero, palautetaan nolla (0).
VAL on STR$-funktion käänteisfunktio.

ESIM:

```
10  LINE INPUT "ANNA IKÄSI :"; A$
20  B%=VAL (A$)
30  PRINT
40  PRINT "OLET" ; B% ; "VUODEN IKÄINEN."
```

Rivillä 10 pyydettävään ikään voit vastata syöttämällä joko esim. luvun "15", merkkijonon "15 VUOTTA" tai merkkijonon "15vuotta". Rivillä 20 syötetystä merkkijonosta poimitaan lukuarvo (numero-osa tulee olla ensimmäisenä) VAL-funktiolla ja saatu arvo talletetaan muuttujaan B%. Rivi 30 tulostaa tyhjän rivin ja rivi 40 tuloksen.

LOOGISET OPERAATTORIT

Loogisilla operaattoreilla suoritetaan muuttujille ja/tai vakioille relaatiotestejä (kahden arvon vertaus), Boolean operaatioita tai bittikäsittelyä. Arvo, jonka operaatio palauttaa (=tulos), on joko TOSI (ei nolla) tai EPÄTOSI (nolla).
Aritmeettiset ja relaatio-operaatiot suoritetaan aina ennen loogisia operaatiota.
BASIC-tulkki, suorittaessaan loogisia operaatioita, muuntaa ensin operandit etumerkeillä varustetuiksi kokonaisluvuksi ja suorittaa sitten operaation bitti kerrallaan alkaen vähitenmerkitsevästä bitistä.
Sekä MSX- että SVI-BASIC sisältää seuraavat loogiset operaatiot:

AND NOT OR XOR EQV IMP

1. AND eli JA

TOTUUSTAULU

X	Y	X AND Y
0	0	0
0	1	0
1	0	0
1	1	1

ESIM:

```
      X= &B0001
      Y= &B1001
X AND Y= &B0001

      X= &B00110011 11110011
      Y= &B10101010 10101010
X AND Y= &B00100010 10100010
```

```
27 AND 12 = 8

        X= &B00000000 00011011  (=27)
        Y= &B00000000 00001100  (=12)
X AND Y= &B00000000 00001000  (=8)

—2 AND 5 = 4

        X= &B11111111 11111110  (=—2)
        Y= &B00000000 00000101  (=5)
X AND Y= &B00000000 00000100  (=4)
```

2. NOT eli EI

TOTUUSTAULU

X	NOT X
0	1
1	0

ESIM:

```
    X= &B0011
NOT X= &B1100

NOT 0 = —1

    X= &B00000000 00000000  (=0)
NOT X= &B11111111 11111111  (=—1)

NOT —6 = 5
    X= &B11111111 11111010  (=—6)
NOT X= &B00000000 00000101  (=5)

NOT 5 = —6
    X= &B00000000 00000101  (=5)
NOT X= &B11111111 11111010  (=—6)
```

NOT X = —(X + 1), koska NOT-operaatio muodostaa argumentin X yhden komplementin (= vastaluku) ja kahden komplementti saadaan lisäämällä yhden komplementtiin yksi (1). Loogisia operaatioita käytetään usein päätöksen teossa, kun ohjelmassa haaraudutaan tai suoritetaan esim. IF-lausetta.

IF NOT C% THEN GOTO 100 ELSE GOTO 200

Jos muuttuja C% < > —1, hypätään riville 100 ja jos muuttujan C% arvo on —1, siirrytään riville 200.

3. OR eli TAI

TOTUUSTAULU

X	Y	X OR Y
0	0	0
0	1	1
1	0	1
1	1	1

ESIM:

```
    X= &B00111100
    Y= &B11100111
X OR Y= &B11111111

—1 OR —2 = —1
    X= &B11111111 11111111  (=—1)
    Y= &B11111111 11111110  (=—2)
X OR Y= &B11111111 11111111  (=—1)
```

4. XOR (EXCLUSIVE OR) eli POISSULKEVA TAI

TOTUUSTAULU

X	Y	X XOR Y
0	0	0
0	1	1
1	0	1
1	1	0

ESIM:

```
    X= &B00000000 10101100
    Y= &B00011000 01011101
X XOR Y= &B00011000 11110001
```

Perusperaatioilla NOT, AND ja OR voidaan toteuttaa mm. XOR-operaatio.

X XOR Y = X AND NOT Y OR Y AND NOT X

5. EQV eli YHTÄKUIN

TOTUUSTAULU

X	Y	X EQV Y
0	0	1
0	1	0
1	0	0
1	1	1

ESIM:

```
        X=  &B00111100 11110000  (=15600)
        Y=  &B10101011 00101011  (=—21717)
X EQV   Y=  &B11010000 00100100  (=26660)
```

6. IMP eli PIENEMPI TAI YHTÄSUURI

TOTUUSTAULU

X	Y	X IMP Y
0	0	1
0	1	1
1	0	0
1	1	1

ESIM:

```
        X=  &B00000000 00110011  (=51)
        Y=  &B00000000 11001111  (=207)
X IMP   Y=  &B11111111 11001111  (=—49)
```

Koska relaatio-operaatioiden tuloksia voidaan käyttää kokonaislukujen tapaan, voidaan XOR, EQV ja IMP -operaatiot toteuttaa seuraavasti:

```
X XOR Y = X <> Y
X EQV Y = X =  Y
X IMP Y = X => Y
```

AND-operaatiota käytetään usein konekielisissä ohjelmissa "maskaamaan" eli tuomaan esille vain yhden tietyn bitin tieto.
OR-operaatiota käytetään yhdistämään kahden tavun tieto yhdeksi tavuksi.

I/O JA ERIKOISFUNKTIOT

I/O -funktioilla tarkoitetaan käskyjä, jotka kohdistuvat tiedon siirtoon koneen ulkoisille oheislaitteille tai päinvastoin. Tässä luvussa ovat mukana myös erikoisfunktiot, joista osa on käytössä vain DISK BASIC-versiossa (levyke-versiossa).

CSRLIN

MUOTO: **CSRLIN**

VERSIOT: **MSX: KASET. + LEVYKE**
SVI: KASET. + LEVYKE

TARKOITUS: Funktio palauttaa kursorin Y-akselin suuntaisen koordinaatin.

LISÄYKSET: Koska sekä MSX- että SVI-BASIC käsittää tekstimoodissa (SCREEN 0) 24 riviä, voi palautettava kokonaisluku saada arvot 0—23.
Vrt. POS-funktioon.

ESIM:

```
10  LOCATE ,2
20  PRINT "OLEMME RIVILLÄ :"; CSRLIN;
30  LOCATE ,10
40  PRINT "OLEMME RIVILLÄ :"; CSRLIN ;
50  LOCATE ,22
60  PRINT "OLEMME RIVILLÄ :"; CSRLIN ;
70  GOTO 70
```

Rivillä 10 asetetaan kursori riville 2 (tosiasiassa rivi 3). Rivillä 20 tulostetaan teksti ja sen perään rivin numero, jolla ollaan (huomio myös puolipiste, joka estää automaattisen tabuloinnin). Näin toimii riviparit 30,40 ja 50,60. Rivillä 70 käytetään GOTO-käskyä päättymättömän silmukan aikaansaamiseen. Ohjelman saa pysähtymään painamalla CTRL+STOP.
Jos rivinumerot halutaan esittää numerosta 1 numeroon 24, tulee CSRLIN-funktiolla palautettavaan arvoon lisätä yksi (esim. PRINT CSRLIN+1).

ERL

MUOTO: **ERL**

VERSIOT: **MSX: KASET. + LEVYKE**
SVI: KASET. + LEVYKE

TARKOITUS: Funktio palauttaa rivinumeron, jolla ohjelmassa on tapahtunut virhe.

LISÄYKSET: Jos virhe tapahtuu komentotilassa, saa ERL arvon 65535.
ERL-funktiota käytetään virheenkäsittelyrutiineissa IF-THEN -lauseessa suorittamaan päätöksen tekoa. Koska virhe voidaan osoittaa tapahtuneen tietyllä rivillä, tiedetään myös virheen laatu ja näin ohjelman kulku pysyy hallinnassa.
Yhtenä esimerkkinä ERL-funktion käytöstä on seuraava, jossa ohjelma itsenäisesti tutkii, onko käytössä KASETTI- vai DISK BASIC.

ESIM:

```
10   ON ERROR GOTO 1000
20   LEVY%=1
30   A$=ATTR$(1)
40   REM   tästä alkaa varsinainen ohjelma
                . . .
990  END
1000 IF ERL=30 THEN LEVY%=0 : RESUME 40
1010 RESUME
```

Rivillä 10 määritellään rivi, jolle hypätään, jos ohjelmassa tapahtuu virhe. Rivillä 20 muuttuja LEVY% saa arvon yksi. Rivillä 30 annetaan käsky, jonka vain DISK BASIC hyväksyy ja jos käytössä on normaali BASIC, aiheutuu tästä virhe. Koska rivillä 10 määrättiin, että virheen tapahtuessa siirrytään riville 1000, siirtyy suoritus virheenkäsittelyrutiiniin riville 1000. Huomaa rivillä 990 oleva END-käsky, jolla estetään ohjelman pääsy virherutiineihin. Rivillä 1000 tutkitaan, onko virhe tapahtunut rivillä 30 ja jos on, annetaan muuttujalle LEVY% arvo 0. Riville 40 palataan RESUME-käskyllä. Muuttujaa LEVY% tutkitaan myöhemmin ohjelmassa ja jos sen arvo on yksi, tiedetään, että käytössä on DISK BASIC. Riippuen muuttujan LEVY% arvosta, voidaan ohjelmassa automaattisesti päätellä, käytetäänkö nauha- vai levytiedostoja.

ERR

MUOTO: **ERR**

VERSIOT: **MSX: KASET. + LEVYKE**
SVI: KASET. + LEVYKE

TARKOITUS: Funktio palauttaa virhekoodin, joka suorituksen aikana on tapahtunut.

LISÄYKSET: Testaamalla funktion ERR palauttamaa arvoa saadaan selville mikä virhe on kyseessä. Kuten ERL-funktion, myös ERR-funktion palauttamaa arvoa tutkitaan yleensä IF-THEN -lauseella.

ESIM:

```
10  ON ERROR GOTO 100
20  PRINT "JAKOLASKU HARJOITUS"
30  PRINT
40  INPUT "ANNA JAETTAVA JA JAKAJA :"; A, B
50  PRINT
60  PRINT "OSAMÄÄRÄ ="; A/B
70  PRINT
80  GOTO 30
100 IF ERR=11 THEN PRINT "NOLLALLA EI VOI JAKAA !":
RESUME 40
110 STOP
```

Rivillä 10 määritellään rivi, jolle siirrytään virheen tapahtuessa. Rivit 30, 50 ja 70 tulostavat tyhjän rivin. Rivillä 20 tulostetaan otsikko. Rivillä 40 syötetään kaksi lukua INPUT-käskyllä. Rivillä 60 tulostuu syötettyjen lukujen osamäärä. Jos jakajaksi on annettu luku nolla, siirtyy suoritus riville 100 (vrt. rivi 10). Rivillä 100 tutkitaan, onko virheenä jako nollalla (vrt. virhekoodi numero 11). Jos näin on, ilmoitetaan tästä ja siirrytään takaisin riville 40 odottamaan uusia lukuja (Huom! siirtyminen virherutiinista takaisin tulee tehdä RESUME-käskyllä). Ohjelman saa pysähtymään painamalla CTRL+STOP.

INP

MUOTO: **INP (<nro>)**

VERSIOT: **MSX: KASET. + LEVYKE**
SVI: KASET. + LEVYKE

TARKOITUS: Funktio lukee I/O -portin <nro>.

LISÄYKSET: Portti <nro> voi saada arvot 0—255, muita ei huomioida.

Z80-mikroprosessori voi lukea/kirjoittaa nk. I/O -sivulle, joka sisältää 256 muistipaikkaa. Tätä osaa muistista kutsutaan usein suorasivuksi ja se sisältää sekä ulkoisten että sisäisten laitteiden ohjausrekisterit.
Käskyn käyttöä tulisi välttää, sillä väärin käytettynä se voi sekoittaa koneen niin, että on katkaistava virta hetkeksi, jolloin tietysti muistissa olevat ohjelmat poistuvat.
HUOM! vaikka INP-funktio vain lukee tietoa, voi sen huolimaton käyttö tuhota ohjelman. Tieto välittyy BC-rekisteriparin kautta. Palautettava luku on kokonaisluku väliltä 0—255.

ESIM:

```
10    A% = INP(&H40)
20    PRINT A%

RUN   <ENTER>
 255
Ok
```

Ohjelma lukee muuttujalle A% arvon portista &H40 (desimaalilukuna 64) rivillä 10. Rivillä 20 arvo tulostetaan.
Palautettavalle arvolle suoritetaan usein AND- tai OR-operaatio, sillä normaalisti vain yksi tietty bitti ratkaisee fyysisen laitteen tilan.

LPOS

MUOTO: **LPOS (X)**

VERSIOT: **MSX: KASET. + LEVYKE**
SVI: KASET. + LEVYKE

TARKOITUS: Funktio palauttaa kirjoitinpuskurin osoittimen aseman, ei kuitenkaan välttämättä kirjoituspään fyysistä paikkaa.

LISÄYKSET: Ilmaisu X on nk. ''dummy''-argumentti, jonka arvolla ei ole merkitystä. X:n tulee kuitenkin olla kokonaisluku.
Funktion avulla voidaan kirjoittimen tulostuksen leveyttä säätää portaattomasti.

ESIM:

```
210   IF LPOS (0)>49 THEN LPRINT CHR$(13);
```

Tässä esimerkissä tutkitaan, onko kirjoittimen kirjoituspää ylittänyt sarakkeen 49 ja jos on, tulostetaan rivinsiirto (CHR$(13)). Muuttamalla luvun 49 paikalle jonkin kokonaisluvun väliltä 1—80, voit säätää tulostuksen leveyden halutuksi.

OUT

MUOTO: **OUT <port>,<data>**

VERSIOT: **MSX: KASET. + LEVYKE**
SVI: KASET. + LEVYKE

TARKOITUS Funktiolla kirjoitetaan I/O -porttiin <port> tieto <data>.

LISÄYKSET: Sekä osoitteen <port> että tiedon <data> tulee olla kokonaisluku väliltä 0—255.
OUT-funktiolla voidaan kirjoittaa suoraan I/O -väylään ja ohjata näin eri oheislaitteita antamalla tarvittaessa ensin ohjaustavu ja tämän jälkeen tieto, <data>. Funktion käytössä tulee olla huolellinen, sillä huolimaton käyttö saattaa johtaa koneen tilaan, josta päästään pois vain katkaisemalla virta hetkeksi. Tällöin tietysti muistissa olevat ohjelmat katoavat.
Funktio on INP-funktion käänteisfunktio (kts. INP).
Esimerkiksi SPECTRAVIDEOn I/O -porttien merkitykset löydät MASKINSPRÅKMANUAL:sta (SV-7004) tai SERVICE/TECHNICAL MANUAL:sta.

ESIM:
```
5100  OUT &H80,100
```
Ilmaisut <port> ja/tai <data> voidaan antaa heksadesimaalilukuina. Esimerkin rivillä 5100 oleva lause kirjoittaa I/O -porttiin &H80 (desimaalilukuna 128) luvun 100.

PEEK

MUOTO: **PEEK (<addr>)**

VERSIOT: **MSX: KASET. + LEVYKE**
SVI: KASET. + LEVYKE

TARKOITUS: Funktio palauttaa tiedon muistipaikan <addr> osoittamasta muistipaikasta.

LISÄYKSET: Osoitteen <addr> tulee olla kokonaisluku väliltä —32768 <= X <= +65535 ja se voidaan antaa desimaali-, heksadesimaali- tai oktaalilukuna.
Sekä MSX- että SVI-BASIC käsittää 32 ktavun BASIC-tulkin, joka sijaitsee osoitteissa 00000—32767 (yht. 32768 tavua, joka yleensä pyöristetään 32 ktavuksi). Tähän muistiosaan EI voida tietoa (ohjelmia ym.) taltioida, ainoastaan lukea PEEK-funktiolla. BASIC-tulkki on taltioitu nk. ROM-muistiin (Read Only Memory) valmistajan toimesta tehtaalla. Osoitteet 32768—65535 sisältävät 32 ktavua RAM-muistia (Random Access Memory), jonne käyttäjän omat ohjelmat taltioituu (MSX-koneissa RAM-muisti on pankissa 1, kun taas SPECTRAVIDEO:ssa RAM-muisti on jaettu kahteen osaan; 32 ktavua on pankissa 0, sivulla 2 ja toinen 32 ktavua on pankissa 2, sivulla 1. Käytännössä tämä eroavaisuus ei BASIC-ohjelmoijalle merkitse mitään.). Osa RAM-muistista on varattu BASIC-tulkin työtilaksi niin, että lopullinen ohjelmoijalle vapaa muistitila on seuraava:

VERSIO	OSOITTEET	YHTEENSÄ
MSX (64 ktav)	32768—61583	28815 tavua
SVI (SV-318)	49152—61967	12815 tavua
SVI (SV-328)	32768—62686	29919 tavua

Seuraavaksi esimerkki ohjelmasta, joka tulostaa kuvaruudulle RAM-muistin sisältöä vapaan muistitilan alusta (tulostukseen on lisätty jälkeenpäin puolipisteen jälkeiset kommentit).

ESIM:
```
10  FOR A%=&H8000 TO &H8042
20  PRINT HEX$(A%);" ";65536!+A%;" ";PEEK(A%)
30  NEXT
```

```
RUN   <ENTER>
8000    32768   0     ;   BASIC- tulkin käytössä.
8001    32769   20    ;   seuraavan rivin osoite =
8002    32770   128   ;   20+128*256 = 32788 (&H8014).
8003    32771   10    ;   rivinumero (saadaan kaavas-
8004    32772   0     ;   ta) 10+0*256 = 10.
8005    32773   130   ;   FOR-käskyn koodi.
8006    32774   32    ;   välilyönti (ASCII-koodi).
8007    32775   65    ;   A-kirjaimen ASCII-koodi.
8008    32776   37    ;   "%"-merkki.
8009    32777   239   ;   FOR-lauseen "="-merkki.
800A    32778   12    ;   tieto heksadesimaaliluvusta.
800B    32779   0     ;   laskurin alkuarvo =
800C    32780   128   ;   0+128*256 = 32768 (&H8000).
800D    32781   32    ;   välilyönti (ASCII-koodi).
800E    32782   217   ;   TO-sanan koodi.
800F    32783   32    ;   välilyönnin ASCII-koodi.
8010    32784   12    ;   tieto heksaluvusta.
8011    32785   66    ;   laskurin loppuarvo =
```

```
8012    32786    128    ;    66+128*256 = 32834 (&H8042).
8013    32787    0      ;    rivin lopetus -merkki.
8014    32788    59     ;    seuraavan rivin osoite =
8015    32789    128    ;    59+128*256 = 32827 (&H803B).
8016    32790    20     ;    rivinumero =
8017    32791    0      ;    20+0*256 = 20.
8018    32792    145    ;    PRINT-käskyn koodi.
        ....
803A    32826    0      ;    edellisen rivin lopetus.
803B    32827    65     ;    seuraavan rivin osoite =
803C    32828    128    ;    65+128*256 = 32833 (&H8041).
803D    32829    30     ;    rivinumero =
803E    32830    0      ;    30+0*256 = 30.
803F    32831    131    ;    NEXT-käskyn koodi.
8040    32832    0      ;    rivin lopetus.
8041    32833    0      ;    tästä alkaisi seuraava rivi,
8042    32834    0      ;    mutta koska käytössä on vain
                        ;    rivit 10—30, ei seuraavaa
                        ;    riviä ole.
```

Huomaa, kuinka BASIC-tulkki merkitsee viimeisen käytössä olevan rivin loppuun (tällä kertaa muistipaikat 32833 ja 32834) kaksi muistipaikkaa nolliksi. Tällä tavoin tulkki kontrolloi BASIC-ohjelman päättyvän. Seuraava ohjelma alkaisi muistipaikasta 32833. Kokeile sitä lisäämällä rivi 100 PRINT "HUUHAA" ja huomaa ero.
Itse ohjelma toimii siten, että rivillä 10 määritellään FOR-lause, jossa kokonaislukumuuttuja A saa arvot 32768 (&H8000) — 32834 (&H8042). Rivillä 20 tulostetaan ensin tutkittavan muistipaikan osoite heksadesimaali muodossa, estetään rivinsiirto puolipisteellä, tulostetaan kaksi tyhjää (" "), sitten sama osoite desimaalimuodossa (muuttujaan A% on lisättävä luku 65536, jotta saataisiin oikea desimaaliluku), taas kaksi tyhjää ja lopuksi muistipaikan sisältö desimaalimuodossa. Rivin 30 NEXT-käsky sulkee silmukan.

POKE

MUOTO: **POKE <addr>,<data>**

VERSIOT: **MSX: KASET. + LEVYKE**
SVI: KASET. + LEVYKE

TARKOITUS: Funktiolla kirjoitetaan RAM-muistin osoitteeseen <addr> tieto <data>.

LISÄYKSET: Osoite <addr> annetaan kokonaislukuna ja se voi olla välillä —32768 — +65535. Jos osoite annetaan negatiivisena lukuna, saadaan todellinen osoite vähentämällä annettu osoite luvusta 65536 (esim. POKE —112,10 = POKE 65424,10).
Talletettava tieto <data> voi olla kokonaisluku väliltä 0—255.
Sekä <addr> että <data> voidaan antaa desimaali-, heksadesimaali- tai oktaalilukuna.
Vain RAM-muistialueelle voidaan kirjoittaa tietoa POKE-käskyllä (kts. PEEK-funktio).

ESIM:

```
10  FOR A=50000 TO 50009
20  PRINT PEEK(A);
30  NEXT
40  PRINT : PRINT
50  FOR A=50000 TO 50009
60  POKE A,77
70  NEXT
80  FOR A=50000 TO 50009
90  PRINT PEEK(A);
100 NEXT
```

Ohjelma muodostuu kolmesta FOR-NEXT -silmukasta: rivit 10—30, 50—70 ja 80—100. Ensimmäinen silmukka tulostaa kuvaruudulle muistipaikkojen 50000—50009 sisällön, seuraava silmukka taltioi vastaaviin muistipaikkoihin luvun 77 ja viimeinen silmukka tulostaa vastaavien muistipaikkojen sisällön POKE-funktion jälkeen. Rivillä 40 tulostetaan yksi (1) tyhjä rivi (HUOM! koska rivillä 20 on PRINT-lauseen lopussa puolipiste, täytyy antaa kaksi PRINT-käskyä, jotta tulostuisi yksi tyhjä rivi. Ensimmäinen PRINT-käsky siirtää kursorin seuraavan rivin alkuun ja jälkimmäinen PRINT-käsky tekee tyhjän rivin).

POS

MUOTO: **POS (X)**

VERSIOT: **MSX: KASET. + LEVYKE**
SVI: KASET. + LEVYKE

TARKOITUS: Funktio palauttaa sarakkeen numeron, jossa kursori sillä hetkellä sijaitsee.

LISÄYKSET: Ilmaisu X on nk. "dummy"-argumentti, eikä sen arvolla ole merkitystä.
Funktio palauttaa arvon väliltä 0—39 tai 0—79, jos käytössä on 80 merkkiä/rivi -laajennin. MSX-BASIC:n WIDTH-käsky ei vaikuta palautettavaan arvoon, vaikka näytön leveys määriteltäisiin esim. 20 merkiksi.
Funktion avulla voidaan mm. ohjata tulostuksen leveyttä kuvaruudulla, josta esimerkki.

ESIM:

```
10  PRINT
20  PRINT "BASIC"; TAB (POS (0)+3);"PASCAL";
30  PRINT TAB (POS (0)+3);"FORTRAN"; TAB (POS (0) +3);"COBOL"
40  PRINT "******"; TAB (POS (0)+3);"*******";
50  PRINT TAB (POS (0)+3);"********"; TAB (POS (0) +3);"******"
60  PRINT
```

Rivit 10 ja 60 tulostavat yhden tyhjän rivin. Rivit 20 ja 30 tulostavat otsikot siten, että ensin tulostuu alkaen vasemmalta sana BASIC, sitten kolme tyhjää (POS (0)+3 -lause siirtää kursoria kolme saraketta oikealle), jne.
Mikäli halutaan säätää tulostuksen leveys kuvaruudulla tiettyyn arvoon, toimitaan seuraavasti:

```
210  IF POS (0)>19 THEN PRINT CHR$(13);
```

Tällä tavoin voidaan säätää tulostus rivillä 20:een sarakkeeseen.

USR

MUOTO: **USR [X] (Y)**

VERSIOT: **MSX: KASET. + LEVYKE**
SVI: KASET. + LEVYKE

TARKOITUS: Funktiolla kutsutaan konekielinen aliohjelma numero X.

LISÄYKSET: Ilmaisu X (=aliohjelman numero) voi olla kokonaisluku väliltä 0—9. Oletusarvona on nolla (0).
Ilmaisulla Y välitetään aliohjelmaan tietoa.
Konekielinen aliohjelma tulee määritellä ennen sen kutsumista. Määrittely suoritetaan DEF USR -käskyllä joko ohjelman alussa tai viimeistään ennen aliohjelman kutsumista. Paluu aliohjelmasta suoritetaan konekielisestä ohjelmasta käsin (RET).
Konekieliset ohjelmat suositellaan ladattaviksi RAM-muistin yläosaan, jolloin alue voidaan BASIC-ohjelman alussa suojata. Suojaus suoritetaan siten, että konekielisten aliohjelmien tarvitsema muistitila lasketaan yhteen ja vähennetään ylimmästä BASIC:n käytössä olevasta osoitteesta (kts. PEEK-funktio). Saatu osoite asetetaan CLEAR-käskyllä BASIC:n ylimmäksi käytettävissä olevaksi osoitteeksi.

ESIM:

```
10  DEF USR5=&H0000
20  PRINT " TÄMÄ OHJELMA TUHOAA"
30  PRINT " ITSE ITSENSÄ, KUN"
40  PRINT "PAINAT JOTAIN NÄPPÄINTÄ."
50  A$=INPUT$(1)
60  A=USR5(0)
```

Rivillä 10 määritellään konekielisen aliohjelman numero viisi (5) alkavan osoitteesta 000000 (tämä on ROM-muistissa oleva RAM-muistin tarkistusohjelma, joka käynnistyy aina kytkettäessä virta koneeseen). Rivit 20—40 tulostavat tekstiä. Rivillä 50 annetaan käsky lukea näppäimistöltä yksi merkki ja kun sellainen annetaan, siirrytään seuraavalle rivillle. Rivillä 60 siirrytään konekieliseen ohjelmaan numero viisi. Tämän esimerkkiohjelman vaikutus on sama, kuin koneesta katkaistaisiin hetkeksi virta.

VARPTR

MUOTO: **VARPTR (<muuttuja>)**
VARPTR (#<nro>)

VERSIOT: **MSC: KASET. + LEVYKE**
SVI: KASET. + LEVYKE

TARKOITUS: Funktio palauttaa osoitteen, johon muuttuja <muuttuja> tai tiedosto numero #<nro> on taltioitu.

LISÄYKSET: Funktio VARPTR (#<nro>) on vain DISK BASIC:n käytössä.
Muuttujalla <muuttuja> tulee olla arvo ennen funktion käyttöä tai tulostuu virheilmoitus "Illegal function call" ja suoritus siirtyy komentotilaan.
Funktiolla VARPTR (<muuttuja>) voidaan tutkia muuttujan, merkkijonon tai taulukon sijaintia RAM-muistissa.
Funktio palauttaa kokonaisluvun, joka on välillä —32768 — +32767. Jos funktio palauttaa negatiivisen luvun, saadaan oikea osoite lisäämällä palautettuun lukuun 65536.
Funktiota käytetään usein ennen konekieliseen aliohjelmaan siirtymistä selvittämään aliohjelmassa käytettävän muuttujan, merkkijonon, taulukon tai tiedoston sijainti muistissa.
Taulukon sijainti (tieto ensimmäisen alkion sijainnista) saadaan antamalla käsky VARPTR (A#(0)), joka palauttaa muuttujan A (tällä kertaa kaksinkertaisen tarkkuuden lukuja sisältävä matriisi) alkion numero 0 sijainnin.
Jos muuttuja <muuttuja> on kokonaisluku ja funktio palauttaa luvun A (jos negatiivinen, lisätään 65536), sisältää

A:n osoittama muistipaikka muuttujan <muuttuja> vähiten merkitsevän tavun ja
A+1:n osoittama muistipaikka eniten merkitsevän tavun.

Kokonaislukumuuttuja varaa siis kaksi (2) muistipaikkaa ja ne talletetaan kahden komplementtimuodossa.
Jos muuttuja <muuttuja> on yksinkertaisen tarkkuuden luku ja funktio palauttaa luvun A (negatiiviseen arvoon lisätään 65536), sisältää

A:n osoittama muistipaikka muuttujan vähiten merkitsevän tavun,
A+1:n osoittama muistipaikka muuttujan toiseksi vähiten merkitsevän tavun,
A+2:n osoittama muistipaikka muuttujan eniten merkitsevän tavun ja
A+3:n osoittama muistipaikka muuttujan eksponentin.

Yksinkertaisen tarkkuuden muuttuja varaa muistia neljä (4) tavua.
Jos muuttuja <muuttuja> on kaksinkertaisen tarkkuuden luku ja funktio palauttaa luvun A (neg. arvoon lisätään luku 65536), sisältää

A:n osoittama muistipaikka muuttujan vähiten merkitsevän tavun,
A+1:n osoittama muistipaikka muuttujan toiseksi vähiten merkitsevän tavun,
...
A+6:n osoittama muistipaikka muuttujan eniten merkitsevän tavun ja
A+7:n osoittama muistipaikka muuttujan eksponentin.

Kaksinkertaisen tarkkuuden muuttuja varaa kahdeksan (8) muistipaikkaa.
Jos muuttuja <muuttuja> on merkkijono ja funktio palauttaa luvun A (neg. arvoon lisätään 65536), sisältää

A:n osoittama muistipaikka muuttujan (merkkijonon) pituuden,
A+1:n osoittama muistipaikka osoitteen vähiten merkitsevän tavun, jonne merkkijono on taltioitu ja
A+2:n osoittama muistipaikka osoitteen eniten merkitsevän tavun, jonne merkkijono on taltioitu.

Merkkijonon ensimmäisen merkin osoite saadaan kaavalla:
(A+1) + (A+2)*256, jossa A+1 ja A+2 ovat edellä määriteltyjen muistipaikkojen sisällöt.

Merkkijono varaa muistia yhtä monta tavua, kuin merkkijonossa on merkkejä + kolme (3) muistipaikka.

ESIM:

```
10   A%=1
20   PRINT VARPTR (A%)+65536!
RUN <ENTER>
 32800
Ok
```

Ohjelma tulostaa rivillä 10 määritellyn muuttujan A% sisällön osoitteen (huom! lukuun on lisätty 65536, sillä muuten tulos olisi negatiivinen osoite). Jos nyt annat käskyn: PRINT

PEEK (32800) ja painat <ENTER>, tulostuu muistipaikan 32800 osoite, joka on muuttujan A% sisältö eli luku 1. Kokeile.

```
PRINT PEEK (32800) <ENTER>
 1
Ok
```

Lisää ohjelmaan rivi

```
30  PRINT
```

ja aja ohjelma uudestaan RUN-käskyllä ja huomaa ero.
BASIC-tulkki taltioi vakiot, muuttujat, merkkijonot ja taulukot aina heti ohjelman perään, kuten edellisessäkin esimerkissä.

WAIT

MUOTO: **WAIT X, <mask> [,<oper>]**

VERSIOT: **MSX: KASET. + LEVYKE**
SVI: KASET. + LEVYKE

TARKOITUS: Funktio lukee I/O -portin numero X arvon tai odottaa luvun <mask> esiintymistä tässä portissa.

LISÄYKSET: Ilmaisulla X tarkoitetaan I/O -portin osoitetta, joka voi olla välillä 0—65535.
Portista X luettavalle tavulle suoritetaan EXOR-operaatio (kts. loogiset operaatiot) ilmaisun <oper> kanssa tai luvun nolla (0), jos ilmaisua <oper> ei anneta. <oper> voi olla kokonaisluku väliltä 0—255.
Luvulle, joka saadaan EXOR-operaation jälkeen, suoritetaan AND-operaatio luvun <mask> kanssa. Ilmaisu <mask> voi olla kokonaisluku välillä 0—255.
Jos saatu tulos on nolla (0), suoritetaan funktio uudestaan ja näin jatketaan, kunnes tulos on ei nolla. Silloin suoritetaan seuraava käsky tai siirrytään ohjelmassa seuraavalle riville.

ESIM: **WAIT &H20,&B00001000,20**

Funktio lukee I/O -portin osoitteesta &H20 (desimaalilukuna 32), suorittaa EXOR-operaation luvun 20 kanssa ja AND-operaation luvun &B00001000 kanssa (testaa, onko luvun bitti kolme eli neljäs bitti "ykkönen"). Jos tulos on < > 0, suoritetaan funktio uudestaan.

BASIC VÄITTÄMÄT JA FUNKTIOT

BEEP

MUOTO: **BEEP**

VERSIOT: **MSX: KASET. + LEVYKE**
SVI: KASET. + LEVYKE

TARKOITUS: Funktiolla tuotetaan "piip"-ääni.

LISÄYKSET: Funktio voidaan korvata "PRINT CHR$ (7)" -lauseella.

ESIM:

```
10  CLS
20  LOCATE 18,11
30  PRINT "     "
40  FOR A=1 TO 300 : NEXT
50  BEEP
60  LOCATE 18,11
70  PRINT "BEEP"
80  FOR A=1 TO 300 : NEXT
90  GOTO 20
```

Ohjelma tulostaa kuvaruudun keskelle noin sekunnin välein tekstin "BEEP" ja tuottaa samanaikaisesti "piip"-äänen. Rivillä 10 tyhjennetään näyttö CLS-käskyllä. Rivillä 20 asetetaan kursori riville 11 sarakkeeseen 18. Rivin 30 PRINT-lause pyyhkii neljä seuraavaa sara-

ketta tyhjäksi. Rivillä 40 on FOR-NEXT -silmukka, joka kuluttaa vain aikaa. Rivillä 50 tuotetaan ääni. Rivillä 60 siirretään kursori takaisin kohtaan 18,11. Rivin 70 PRINT-lause tulostaa tekstin "BEEP". Rivillä 80 vietetään taas aikaa FOR-NEXT -silmukassa. Rivillä 90 annetaan käsky palata takaisin riville 20.

BLOAD

MUOTO: **BLOAD " [<dev>] [<nimi>]"[,R] [,S] [,<offset>]**

VERSIOT: **MSX: KASET. + LEVYKE**
SVI: KASET. + LEVYKE

TARKOITUS: Funktio lataa konekielisen ohjelman I/O -laitteelta <dev>.

LISÄYKSET: <dev> = oheislaite, jolta lataus suoritetaan. Voi olla jokin seuraavista:

CAS: ; datanauhuri (SPECTRAVIDEO) tai normaali kasettinauhuri (MSX).
1: ; levykeasema numero yksi (1). Käsky toimii vain SPECTRAVIDEO:n DISK-BASIC:ssä.
2: ; levykeasema numero kaksi (2). Käsky toimii vain SPECTRAVIDEO:n DISK BASIC:ssä ja koneeseen tulee olla liitettynä kaksi levykeasemaa.
A: ; levykeasema yksi. Käsky toimii vain MSX DISK BASIC:ssä.
B: ; levykeasema kaksi. Käsky toimii vain MSX DISK BASIC:ssä

Jos ilmaisu <dev> jätetään antamatta, olettaa SVI-BASIC laitteeksi datanauhurin (sekä KASETTI että DISK BASIC-versio) ja MSX-BASIC nauhurin (KASETTI-BASIC) tai oletuslevykeaseman (DISK BASIC).
<nimi> = ladattavan ohjelman nimi. Ohjelmalle annetaan nimi BSAVE-funktiolla ohjelmaa taltioidessa. Ladattaessa ohjelmaa nauhalta saattaa kuvaruudulle tulostua teksti "Skip: nimi", jolloin kone ilmoittaa ohittavansa nauhalla olevan ohjelman nimeltä "nimi". Jos halutaan käynnistää ohjelma heti latauksen päätyttyä, lisätään käskyyn kirjain "r" tai "R" pilkulla erotettuna.
Lisäämällä ilmaisu ",S", latautuu konekielitiedosto video-RAM:iin. Tiedosto (kuva) on aikaisemmin taltioitu BSAVE-käskyllä.
Konekielinen ohjelma latautuu osoitteisiin, jotka on määrätty BSAVE-funktion yhteydessä. Antamalla käskyssä ilmaisu <offset> , lisätään aloitus- ja lopetusosoitteita ilmaisun verran. Vain MSX-BASIC.
Jos ilmaisu <nimi> jätetään antamatta (HUOM! mahdollinen vain KASETTI-versiossa), latautuu seuraava (ensimmäinen ohjelma, joka löytyy) ohjelma.

ESIM.

BLOAD "sasa",R ; lataa ja käynnistää ohjelman datanauhurilta (SVI-BASIC tai MSX-BASIC:n KASETTI-versio) tai levykkeeltä (MSX-DISK-BASIC), nimeltä "sasa".

BLOAD "A:Test",r,1000 ; lataa levykeasemasta 1 (A) ohjelman nimeltä "Test, lisäten aloitusosoitetta 1000:lla. (Vain MSX-DISK-BASIC!).

BSAVE

MUOTO: **BSAVE "[<dev>] [<nimi>]",<alku>,<loppu>[,<start>] [,S]**

VERSIOT: **MSX: KASET. + LEVYKE**
SVI: KASET. + LEVYKE

TARKOITUS: Funktiolla tallennetaan oheislaitteelle <dev> konekielinen ohjelma tai muistin osa.

LISÄYKSET: <dev> = oheislaite, jolle ohjelma tai muistin osa talletetaan. Voi olla jokin seuraavista:

CAS: ; datanauhuri (SPECTRAVIDEO) tai normaali kasettinauhuri (MSX).
1: ; levykeasema numero yksi (1). Käsky toimii vain SPECTRAVIDEO:n DISK BASIC:ssä.
2: ; levykeasema numero kaksi (2). Käsky toimii vain SPECTRAVIDEO:n DISK BASIC:ssä ja koneeseen tulee olla liitettynä kaksi levykeasemaa.
A: ; levykeasema yksi. Käsky toimii vain MSX DISK BASIC:ssä.
B: ; levykeasema kaksi. Käsky toimii vain MSX DISK BASIC:ssä.

Jos ilmaisu <dev> jätetään antamatta, olettaa SVI-BASIC laitteeksi datanauhurin (sekä KASETTI että DISK BASIC-versio) ja MSX-BASIC nauhurin (KASETTI-BASIC) tai oletuslevykeaseman (DISK-BASIC).

<nimi> = ohjelman nimi, joka taltioituu nauhalle tai levykkeelle ohjelman mukana. Käytettäessä myöhemmin latausfunktiota BLOAD, verrataan ilmaisua <nimi> funktiolla annettuun nimeen. Ilmaisu voidaan jättää antamatta (KASETTI-versio), jolloin ohjelma tallentuu ilman nimeä.
Ilmaisuilla <alku> ja <loppu> annetaan ohjelman sijainti muistissa. Tallennus alkaa osoitteesta <alku> ja loppuu osoitteeseen <loppu>.
Ilmaisulla <start> voidaan antaa osoite, josta konekieliohjelma käynnistetään. Jos ilmaisua ei anneta, oletetaan ohjelman käynnistyvän osoitteesta <alku>.
Ilmaisulla ",S" taltioidaan video-RAM:n sisältö (kuva) tiedostona nauhalle/levykkeelle. Kts. BLOAD-käsky.

ESIM: **BSAVE "CAS:tetra",&H8800,&H8900,&H88A0**

Funktio tallentaa kasetille ohjelman nimeltä "tetra", alkaen osoitteesta &H8800, osoitteeseen &H8900 asti. Lisäksi talletetaan tieto, että ohjelma käynnistyy osoitteesta &H88A0.

CLOAD

MUOTO: **CLOAD ["<nimi>"]**

VERSIOT: **MSX: KASETTI**
SVI: KASETTI

TARKOITUS: Funktio lataa BASIC-ohjelman nauhurilta muistiin.

LISÄYKSET: Ilmaisulla <nimi> tarkoitetaan ohjelman nimeä, joka annetaan ohjelmalle CSAVE-funktiolla.
Antamalla komento CLOAD latautuu nauhalta seuraava (ensimmäinen ohjelma, joka löytyy) ohjelma.
Funktio tulostaa kuvaruutuun tekstin "Press play on tape" (SVI-BASIC), jolla pyydetään käynnistämään nauhuri PLAY-painikkeesta. Jos PLAY-painike on alhaalla ennen CLOAD-käskyn antamista, ei "Press play on tape" -tekstiä tulostu.
MSX-BASIC:ssä on käytössä kaksi eri lataus/tallennusnopeutta: 1200 ja 2400 baudia (=bittiä/sekunti). Nopeutta voidaan vaihtaa SCREEN-käskyllä.

ESIM:

```
CLOAD                ; lataa ensimmäisen löytyvän ohjelman.
cload "bingo.bas"    ; lataa ohjelman nimeltä "bingo.bas".
```

Funktio poistaa automaattisesti vanhan ohjelman muistista ennen uuden latausta (kirjoittaa uuden ohjelman vanhan ohjelman päälle).

CLOAD?

MUOTO: **CLOAD? [<nimi>]**

VERSIOT: **MSX: KASETTI**
SVI: KASETTI

TARKOITUS: Funktion avulla verrataan muistissa ja kasetilla olevia ohjelmia keskenään.

LISÄYKSET: <nimi> = kasetilla olevan verrattavan ohjelman nimi. Jos ilmaisua ei anneta, suoritetaan vertailu ensimmäisen kasetilta löytyvän ja muistissa olevan ohjelman kanssa.
Jos vertailussa löytyy ero tai eroja, tulostuu ilmoitus "Verify error".

ESIM:

```
CLOAD? "rescue"      ; vertaa muistissa olevaa ohjelmaa kasetilla olevan ohjel-
                       man nimeltä "rescue" kanssa.
```

On huomattava kelata nauha oikeaan kohtaan (ohjelman alkuun) ennen käskyn antamista.

COPY

MUOTO: **MSX: COPY "<file1>"TO "<file2>"**
SVI: COPY <nro1> FROM "<nro2>:<nimi>"

VERSIOT **MSX: LEVYKE**
SVI: LEVYKE

TARKOITUS: Funktio kopioi tiedoston tai ohjelman levykeasemasta (SVI) tai levykkeeltä (MSX) toiselle.

LISÄYKSET: <nimi> = kopioitavan ohjelman tai tiedoston nimi, joka tulee antaa täydellisenä. Vain SVI DISK BASIC.

<nro1> = levykeasema, jolle kopioitava ohjelma taltioidaan (yksi, 1 tai kaksi, 2). Vain SVI.
<nro2> = levykeasema, jolta kopioitava ohjelma ladataan (yksi, 1 tai kaksi, 2). Vain SVI.
<file1> = ilmaisulla annetaan tiedoston nimi ja levykeasema, jolta taltioitava ohjelma ladataan. Vain MSX.
<file2> = ilmaisulla annetaan tiedoston nimi ja levykeasema, jolle kopioitava ohjelma taltioidaan. Vain MSX.
Ilmaisut <file1> ja <file2> ovat seuraavaa muotoa:

<dev> : <name>

Ilmaisu <dev> voi olla "A" (levykeasema yksi) tai "B" (levykeasema kaksi), mutta voidaan jättää myös antamatta, jolloin levykeasemana käytetään oletusarvoasemaa (viimeksi käytössä ollutta asemaa). Ilmaisulla <name> annetaan kopioitavan ohjelman/ohjelmien nimet (apuna voidaan käyttää merkkejä "?" ja "*", joilla korvataan useamman ohjelman nimi). Kts. tiedostokäskyt.

ESIM:

COPY 2 FROM "1:plist" ; kopioi levykeasemasta yksi ohjelman nimeltä "plist" levykeasemaan kaksi (SVI DISK BASIC).

COPY "TEST" TO "KOE"; kopioi oletusarvoasemassa olevan ohjelman nimeltä "TEST" ohjelmaksi nimeltä "KOE". EI poista ohjelmaa "TEST". Vain MSX.

CSAVE

MUOTO: **CSAVE "<nimi>" [,<baud>]**

VERSIOT: **MSX: KASET. + LEVYKE**
SVI: KASET. + LEVYKE

TARKOITUS: Funktio tallentaa muistissa olevan BASIC-ohjelman kasetille.

LISÄYKSET: Ilmaisulla <nimi> annetaan ohjelmalle nimi, joka taltioituu ohjelmakoodin mukana nauhalle. CLOAD-käskyllä voidaan myöhemmin löytää oikea ohjelma annetun nimen perusteella.
Jos ohjelma tai muistissa oleva tiedosto halutaan taltioida ASCII-muodossa (jotta talletettua tiedostoa voitaisiin hyödyntää esim. MERGE-käskyllä), tulee käyttää SAVE-funktiota.
Ilmaisulla <baud> voidaan määritellä ohjelman tallennusnopeus joko 1200 baudiksi (ilmaisun arvo yksi, 1) tai 2400 (ilmaisun arvo kaksi, 2). Käskyllä määritelty tallennusnopeus asettuu automaattisesti oikeaan arvoon CLOAD-käskyllä ohjelmaa muistiin ladatessa. Tallennusnopeus voidaan säätää myös SCREEN-käskyllä. Ilmaisua voidaan käyttää vain MSX-BASIC:ssä.
CSAVE-funktio tallettaa ohjelman koodattuna (puristettuna) binaarimuodossa.

ESIM:

CSAVE "HARJ.BAS" ; tallettaa muistissa olevan ohjelman kasetille nimellä "HARJ.BAS".

Funktio ei poista latauksen jälkeen ohjelmaa muistista.

KEY

MUOTO: **KEY <nro>,"<string>" [+CHR$(X)]**

VERSIOT: **MSX: KASET. + LEVYKE**
SVI: KASET. + LEVYKE

TARKOITUS: Funktiolla asetetaan halutulle funktionäppäimelle (1—10) merkit, jotka automaattisesti kirjoitetaan näppäintä painettaessa.

LISÄYKSET: <nro> = funktionäppäimen numero väliltä 1—10 (huomioi, että funktionäppäimet 6—10 saat painamalla SHIFT+funktionäppäin).
<string> = ilmaisulla annetaan merkit, jotka funktionäppäin tulostaa sitä painettaessa. Usein merkitys on jokin BASIC:n käskyistä. Merkit erotetaan lainausmerkeillä, joista jälkimmäinen on merkitön, jos ohjauskoodia CHR$(X) ei käytetä. Maksimi merkkien määrä on 15.
Ilmaisulla '+CHR$(X)' voidaan lisätä ohjainkoodeja, kuten esim. CHR$(13), joka vastaa <ENTER>-näppäimen painallusta. Ohjainkoodit eivät tulostu KEY LIST -käskyllä, mutta näkyvät yhtenä tyhjänä merkkinä ja varaavat myös yhden merkin tilaa/ohjainkoodi.

ESIM: **KEY 2,CHR$(12)+"LIST"+CHR$(13)**

Lause antaa funktionäppäimelle numero 2 seuraavat toiminnot: CLS (CHR$(12)), LIST (normaali LIST-komento) ja <ENTER> (<ENTER>:n painallusta vastaava ohjainkoodi).
Funktionäppäimet voidaan ohjelmoida kaikki kerralla vaikkapa seuraavasti:

```
10  FOR A% = 1 TO 10
20  B$ = "COLOR"
30  KEY A%,B$
40  NEXT
```

Rivit 10—40 muodostavat FOR-NEXT -silmukan, jonka aikana (rivillä 20) määritellään funktionäppäimen merkitykseksi teksti (käsky) COLOR ja jonka jokainen funktionäppäin vuorollaan saa (rivi 30).
Funktionäppäinten merkitykset voidaan helposti poistaa seuraavasti:

```
10  FOR A% = 1 TO 10
20  KEY A%," "
30  NEXT
```

SVI-BASIC:ssä voidaan funktionäppäinten alkuperäiset merkitykset palauttaa (ilman, että tarvitsee katkaista virtaa ON/OFF -kytkimestä) seuraavasti:

DEFUSR = 89 : A = USR(0) <ENTER>

MSX-BASIC:ssä saadaan funktionäppäimet pois näkyvistä (näppäinten merkitys säilyy) seuraavasti:

DEFUSR = &HCC : A = USR(0) <ENTER>

ja takaisin näkyviin:

DEFUSR = &HCF : A = USR(0) <ENTER>

DEFUSR-funktiot ovat konekielisten aliohjelmien kutsuja (kts. DEF USR).

```
10  PRINT "PAINA JOTAIN FUNKTIONÄP-"
20  PRINT "PÄINTÄ JA TÄMÄN OHJELMAN"
30  PRINT " LISTAUKSEN."
40  A$ = "LIST" + CHR$(13)
50  FOR B% = 1 TO 10
60  KEY B%,A$
70  NEXT
80  A$ = INKEY$ : IF A$ = "" THEN GOTO 80
90  IF A$ <> "LIST" THEN GOTO 80
100 LIST
```

Rivit 10—30 tulostavat otsikon. Rivillä 40 määritellään muuttujalle A$ merkitys, joka rivillä 60 annetaan jokaiselle funktionäppäimelle FOR-NEXT -silmukan aikana (rivit 50—70). Rivillä 80 odotetaan jonkin näppäimen painallusta. Kun jotain näppäintä painetaan, siirrytään riville 100 ja suoritetaan normaali ohjelman listaus.

KEY LIST

MUOTO: **KEY LIST**

VERSIOT: **MSX: KASET. + LEVYKE**
SVI: KASET. + LEVYKE

TARKOITUS: Funktiolla tulostetaan funktionäppäinten merkitykset kuvaruudulle järjestyksessä.

LISÄYKSET: Funktio tulostaa kaikki näppäinten merkit (15 kpl/näppäin). Ohjainkoodit (CHR$(X) -funktiot) tulostuvat tyhjinä (välilyönteinä).

ESIM: **MSX-BASIC**

```
KEY LIST <ENTER>
color
auto
goto
list
run
color 15,4,4
cload"
cont
list.
  run (HUOM! "run"-sanaa edeltävä tyhjä, joka tulee CHR$(12)-koodista eli CLS.)
Ok
```

SVI-BASIC

```
KEY LIST <ENTER>
color       auto
goto        list
run         color 15,4,5
cload"      cont
list         run
Ok
```

KEY ON/OFF

MUOTO:
KEY ON
KEY OFF

VERSIOT:
MSX: KASET. + LEVYKE
SVI: funktiolla eri merkitys.

TARKOITUS: Funktio asettaa kuvaruudun alareunassa näkyvät funktionäppäinten merkitykset näkyviin tai pois näkyvistä.

LISÄYKSET: Funktiolla KEY ON asetetaan näppäinten merkitykset näkyviin ja funktiolla KEY OFF pois näkyvistä (samalla vapautuu kuvaruudun alin rivi, rivi 24, BASIC:n käyttöön).
HUOM! funktiolla on eri merkitys SVI-BASIC:ssä, jossa KEY-funktiolla voidaan estää tai sallia funktionäppäinten aiheuttamat keskeytykset (kts. keskeytyspalvelukäskyt).

ESIM:

```
10  PRINT "PAINAMALLA JOTAIN NÄPPÄINTÄ"
20  PRINT "SAAT FUNKTIONÄPPÄINTEN MER-
30  PRINT "KITYKSET NÄKYVIIN JA POIS."
40  A$=INPUT$(1)
50  KEY OFF
60  A$=INPUT$(1)
70  KEY ON
80  GOTO 40
```

Rivit 10—30 tulostavat ohjeet. Rivillä 40 ohjelman suoritus pysähtyy ja jää odottamaan näppäimistöltä yhtä merkkiä. Kun jotain näppäintä painetaan, siirrytään riville 50, jolla poistetaan näppäimet näytöltä. Rivillä 60 jäädään taas odottamaan yhtä merkkiä näppäimistöltä ja rivillä 70 asetetaan näppäimet näkyviin. Ohjelma on ikuinen silmukka, sillä rivillä 80 palataan takaisin riville 40 GOTO-käskyllä odottamaan jonkin näppäimen painallusta.

KILL

MUOTO:
MSX: KILL "[<dev> :] <nimi>"
SVI: KILL "<nro>:<nimi>"

VERSIOT:
MSX: LEVYKE
SVI: LEVYKE

TARKOITUS: Funktio poistaa levykkeeltä ohjelman tai tiedoston nimeltä <nimi>.

LISÄYKSET: SVI-BASIC:ssä voidaan määritellä levykeasema, jolta ohjelma tai tiedosto poistetaan ilmaisulla <nro> (1 tai 2).
MSX-BASIC:ssä ilmaisulla <dev> annetaan myös levykeasema, jolta ohjelma poistetaan. Kirjain "A" vastaa asemaa yksi ja kirjain "B" asemaa kaksi. Jättämällä ilmaisu antamatta, poistetaan ohjelma/tiedosto oletusarvoasemasta.
Ilmaisussa <nimi> voidaan MSX DISK BASIC:ssä käyttää merkkejä "?" ja "*", joiden avulla voidaan poistaa kerralla useita ohjelmia/tiedostoja. Kts. tiedostokäskyt.
Tiedostoa ei voida levykkeeltä poistaa KILL-funktiolla, jos tiedosto on avoin (kts. OPEN-käsky).

ESIM:

```
KILL "TESTI.BAS"    ; poistaa oletusarvoasemasta ohjelman nimeltä "TESTI.BAS"
                      (vain MSX DISK BASIC).
KILL "2:PELI1"      ; poistaa levykeasemasta kaksi (2) ohjelman nimeltä "PELI1"
                      (vain SVI DISK BASIC).
```

LOAD

MUOTO: **LOAD "[<dev> :] [<nimi>]" [,R]**

VERSIOT: **MSX: KASET. + LEVYKE**
SVI: KASET. + LEVYKE

TARKOITUS: Funktio lataa muistiin BASIC-ohjelman tai tiedoston nimeltä <nimi>.

LISÄYKSET: <dev> = oheislaite, jolta ohjelma tai tiedosto ladataan, voi olla jokin seuraavista:

CAS	;	datanauhuri (SPECTRAVIDEO) tai normaali kasettinauhuri (MSX).
1	;	levykeasema numero yksi (1). Vain SVI.
2	;	levykeasema numero kaksi (2). Vain SVI (ja koneeseen tulee olla liitettynä kaksi levykeasemaa).
A	;	levykeasema yksi (1). Vain MSX.
B	;	levykeasema kaksi (2). Vain MSX.

Jos ilmaisu <dev> jätetään antamatta, olettaa SVI-BASIC laitteeksi datanauhurin (sekä KASETTI että DISK BASIC-versio) ja MSX-BASIC nauhurin (KASETTI-BASIC) tai oletusarvolevykeaseman (DISK BASIC).
<nimi> = ohjelman tai tiedoston nimi, joka ladataan. Ilmaisu voidaan jättää antamatta (mahdollinen vain KASETTI-versiossa), jolloin muistiin latautuu seuraava kasetilta löytyvä tiedosto (tiedoston tulee olla ASCII-tiedosto eli talletettu SAVE-komennolla ASCII-muotoisena).
LOAD-funktio sulkee kaikki avoinna olevat tiedostot ja poistaa muistista edellisen ohjelman.
Ilmaisulla "R" (tai "r", pilkulla erotettuna) voidaan ohjelma käynnistää automaattisesti latauksen päätyttyä, jolloin myös avoinna olevia tiedostoja EI suljeta, vaan ne jäävät ladattavan ohjelman käyttöön.

ESIM:

LOAD "CAS:testi"	;	lataa kasetilta ohjelman nimeltä "testi".
LOAD "testi",R	;	lataa kasetilta (SVI-BASIC) tai levykkeeltä (MSX DISK BASIC) ohjelman nimeltä "testi", joka latauksen jälkeen käynnistyy automaattisesti.
LOAD "1:plist",r	;	lataa levykeasemasta 1 ohjelman "plist", joka samalla käynnistetään (HUOM! toimii vain SVI DISK BASIC:ssä).

MERGE

MUOTO: **MERGE "[<dev> :] [<nimi>]"**

VERSIOT: **MSX: KASET. + LEVYKE**
SVI: KASET. + LEVYKE

TARKOITUS: Funktiolla yhdistetään ASCII-tiedoston ja muistissa olevat ohjelmarivit keskenään.

LISÄYKSET: <dev> = oheislaite, jolta ASCII-tiedosto muistiin ladataan, voi olla jokin seuraavista:

CAS	;	datanauhuri (SPECTRAVIDEO) tai normaali kasettinauhuri (MSX).
1	;	levykeasema numero yksi (1). Vain SVI.
2	;	levykeasema numero kaksi (2). Vain SVI (ja koneeseen tulee olla liitettynä kaksi levykeasemaa).
A	;	levykeasema yksi (1). Vain MSX.
B	;	levykeasema kaksi (2). Vain MSX.

Jos ilmaisu <dev> jätetään antamatta, olettaa SVI-BASIC laitteeksi datanauhurin (sekä KASETTI että DISK BASIC-versio) ja MSX-BASIC nauhurin (KASETTI-BASIC) tai oletusarvolevykeaseman (DISK BASIC).
<nimi> = ladattavan tiedoston nimi. Jos <nimi> jätetään antamatta (mahdollista vain KASETTI-versiossa), latautuu nauhalta seuraava ASCII-tiedosto, joka löytyy.
MERGE-funktiota käytetään silloin, kun ohjelma ei kerralla mahdu muistiin. Ohjelman kuluessa ladataan oheismuistista (kasetilta tai levykkeeltä) nk. ohjelmamoduleita, jotka ovat ASCII-muodossa tallennettuja BASIC-ohjelmia (kts. SAVE-funktio).
Jos muistissa ja MERGE-käskyllä ladattavassa ASCII-tiedostossa on samoja rivinumeroita, korvaa MERGE-funktiolla ladattavat rivit muistissa olevat. MERGE-käskyn jälkeen palataan aina komentotilaan.

ESIM:

```
...
245 PRINT "ODOTA HETKI, LATAAN AAKKOSTUSOHJELMAN."
250 KEY 5,"GOTO 265" + CHR$(13)
```

```
255 PRINT "OHJELMA JATKUU FUNKTIONÄPPÄIMESTÄ 5."
260 MERGE "CAS:AAKKOS"
265 REM ohjelma AAKKOS korvaa rivit 265—1200
```

Rivillä 245 tulostetaan teksti latauksesta. Rivillä 250 ohjelmoidaan funktionäppäimelle numero 5 merkitys "GOTO 265", jotta ohjelmaa voidaan jatkaa kätevästi MERGE-funktion jälkeen (MERGE:n jälkeen palataan komentotilaan ja tästä syystä on käyttäjän joskus jopa mahdotonta tietää, miltä riviltä pitäisi suoritusta jatkaa, sillä RUN-komento aloittaisi ohjelman alusta). Rivillä 250 tulostetaan ohje ohjelman jatkamisesta funktionäppäimellä 5 MERGE:n jälkeen. Rivillä 260 ladataan seuraava ohjelmamoduli kasetilta muistiin. Rivin 265 REM-lause poistuu muistista latauksen jälkeen, jos ladattavan ohjelman "AAKKOS" rivit alkavat numerosta 265.

MOTOR

MUOTO: **MOTOR ON**
MOTOR OFF

VERSIOT: **MSX: KASET. + LEVYKE**
SVI: KASET. + LEVYKE

TARKOITUS: Funktiolla ohjataan kasettiaseman moottoria.

LISÄYKSET: Komennolla MOTOR ON voidaan nauhurin moottori käynnistää ja MOTOR OFF -komennolla moottori pysäyttää. Jos käytössä on SVI-BASIC (ja datanauhuri SV-904/903), on nk. REMOTE-ohjaussignaali aina kytkettynä, kun taas MSX-BASIC:ssä tulee koneen ja nauhurin välisen kaapelin erillisen REMOTE-johdon kytkentä varmistaa.
Jos nauhurin pyöriessä annetaan MOTOR OFF -komento, pysähtyy moottori. Moottori voidaan vastaavasti käynnistää MOTOR ON -komennolla uudestaan. Jos MOTOR ON -komento annetaan moottorin jo pyöriessä, ei komennolla ole vaikutusta. Pelkkä MOTOR-komento kääntää moottoria ohjaavan kytkimen vastakkaiseen asentoon.

ESIM: **MOTOR** ; joko käynnistää tai sammuttaa moottorin, riippuen moottoria ohjaavan "kytkimen" tilasta.

SAVE

MUOTO: **SAVE "[<dev> :] [<nimi>] "[,A]**

VERSIOT: **MSX: KASET. + LEVYKE**
SVI: KASET. + LEVYKE

TARKOITUS: Funktiolla talletetaan ohjelma joko nauhalle tai levykkeelle.

LISÄYKSET: <dev> = oheislaite, jolle ohjelma talletetaan, voi olla jokin seuraavista:

CAS ; datanauhuri (SPECTRAVIDEO) tai normaali kasettinauhuri (MSX).

1 ; levykeasema numero yksi (1). Vain SVI.

2 ; levykeasema numero kaksi (2). Vain SVI (ja koneeseen tulee olla liitettynä kaksi levykeasemaa).

A : levykeasema yksi (1). Vain MSX.

B ; levykeasema kaksi (2). Vain MSX.

Jos ilmaisu <dev> jätetään antamatta, olettaa SVI-BASIC laitteeksi datanauhurin (sekä KASETTI että DISK BASIC -versio) ja MSX-BASIC nauhurin (KASETTI-BASIC) tai oletusarvolevykeaseman (DISK BASIC).
<nimi> = ohjelman nimi, joka taltioidaan nauhalle tai levykkeelle ohjelman mukana. Ilmaisu <nimi> voidaan jättää antamatta (mahdollista vain KASETTI-versiossa), jolloin ohjelma taltioituu ilman nimeä.
Jos levykkeellä on saman niminen tiedosto, jota SAVE-käskyllä ollaan taltioimassa, korvaa uusi tiedosto vanhan.
Mikäli ohjelma halutaan taltioida ASCII-muodossa (välttämätöntä, jos ohjelma halutaan myöhemmin yhdistää MERGE-funktiolla), lisätään SAVE-lauseen perään ilmaisu ",A".
FILES-komennolla tulostuu ASCII-tiedostojen nimen perään tähti (*).

ESIM: **SAVE "CAS:"** ; käsky taltioi muistissa olevan ohjelman nauhalle ilman nimeä.

SAVE "1:test",A ; taltioi muistissa olevan ohjelman levykeasemaan yksi (toimii vain SVI DISK BASIC:ssä) ASCII-muodossa nimellä "test".

SOUND

MUOTO: **SOUND ON**
SOUND OFF

VERSIOT: **MSX: ei käytössä**
SVI: KASET. + LEVYKE

TARKOITUS: Funktio kytkee nauhurin audio-kanavan päälle/pois päältä.

LISÄYKSET: Komento SOUND ON kytkee äänikanavan päälle ja komento SOUND OFF kanavan pois päältä.
Koska SPECTRAVIDEO:n datanauhuri (SV-904/903) on kaksikanavainen, voidaan sen avulla kuunnella myös normaaleja äänitteitä C-kasetilta. Tällöin ääni kulkeutuu koneen kautta television tai monitorin audio-kanavaan. Koska nauhurit SV-904/903 ovat tarkoitettu datanauhureiksi, tulee niiden käyttöä C-kasettien kuunteluun välttää. Ominaisuutta voidaan kuitenkin hyödyntää audio-visuaalisesti siten, että ohjelma tulostaa kuvaruutuun tekstiä/grafiikkaa ja nauhalta tulee asiaan kuuluvaa esim. selostusta.

ESIM:

```
10  PRINT "LAITA NAUHURIIN JOKIN VALMIIKSI"
20  PRINT "ÄÄNITETTY NAUHA JA PAINA PLAY- "
30  PRINT "NAMIKKA ALAS. PAINA SITTEN NÄP- "
40  PRINT "PÄIMISTÖLTÄ MITÄ TAHANSA NÄP.."
50  A$ =INPUT$(1)
60  MOTOR ON : SOUND ON
70  PRINT "KUN ET HALUA KUUNNELLA ENEMPÄÄ"
80  PRINT " PAINA JOTAIN NÄPPÄINTÄ."
90  A$ =INPUT$(1)
100 SOUND OFF
110 PRINT "JOS HALUAT ÄÄNEN TAKAISIN"
120 PRINT " PAINA JOTAIN NÄPPÄINTÄ."
130 GOTO 60
```

Rivit 10—40 tulostavat ohjeita. Rivillä 50 odotetaan näppäimistöltä yhden merkin painamista. Rivillä 60 käynnistetään nauhurin moottori (jollei se jo pyöri) ja avataan äänikanava. Rivit 70 ja 80 tulostavat ohjeita ja rivillä 90 odotetaan yhden merkin painamista. Rivillä 100 kytketään nauhurilta tuleva ääni pois. Rivit 110 ja 120 tulostavat ohjeita ja rivillä 130 hypätään takaisin riville 60.

KESKEYTYSPALVELUKÄSKYT

Kaikki 'ON keskeyt.muoto GOSUB/GOTO' -käskyt toimivat, kuten GOSUB/GOTO-aliohjelmakäskyt. Ainoana erona on, että käsky siirtyä aliohjelmaan annetaan keskeytyksellä ulkoisesti tai sisäisesti, eikä ohjelmallisesti. Toisin sanoen hypyn aliohjelmaan aiheuttaa keskeytys. Keskeytyksellä tarkoitetaan keskusyksikölle eli mikroprosessorille annettua tietoa siitä, että jotain tärkeää, mitä EI voida sivuuttaa, on tapahtunut. Paluu keskeytysaliohjelmasta tapahtuu, kuten paluu normaalista aliohjelmasta, RETURN-käskyllä (GOSUB...).

ON ERROR GOTO

MUOTO: **ON ERROR GOTO <nro>**

VERSIOT: **MSX: KASET. + LEVYKE**
SVI: KASET. + LEVYKE

TARKOITUS: Käsky määrittelee rivin, jolta suoritus virheen tapahduttua jatkuu.

LISÄYKSET: <nro> = virheenkäsittelyrutiinin ensimmäinen rivinumero, jolle siirrytään virheen jälkeen (kokonaisluku väliltä 0-65529).
Tämän määrittelyn jälkeen aiheuttavat kaikki virheet hypyn riville numero <nro>. Määrittely on voimassa, kunnes annetaan 'ON ERROR GOTO 0' -käsky. Tämän jälkeen tulostuu normaali virheilmoitus ja suoritus siirtyy komentotilaan. Näin käy myös silloin, kun virheenkäsittelyrutiinissa tapahtuu virhe.
Virheenkäsittelyrutiinista palataan aina RESUME-käskyllä.
Funktio toimii myös komentotilassa, joten sillä voidaan määritellä rivi, jolle hypätään annettaessa väärä käsky komentotilasta.

ESIM:

```
10  ON ERROR GOTO 100
20  PRINT "ARVAA MITÄ KOKONAISLUKUA"
30  PRINT " AJATTELIN VÄLILTÄ 1—10."
40  INPUT A%
50  B%=1+INT (RND (-TIME)*10)
60  IF B%>A% THEN ERROR 200 ELSE IF A%>B%
THEN ERROR 201
70  PRINT "ARVASIT OIKEIN!!!"
80  END
90  REM virherutiini
100 IF ERR=200 THEN PRINT "LIIAN PIENI..."
ELSE PRINT "LIIAN ISO..."
110 RESUME 40
```

Rivillä 10 määritellään rivi, jolle siirrytään virheen tapahtuessa. Rivit 20 ja 30 tulostavat tekstiä. Rivillä 40 pyydetään syöttämään jokin luku. Rivillä 50 kone arpoo luvun väliltä 1—10 käyttäen RND-funktiota ja TIME-muuttujaa. Rivillä 60 tarkistetaan, onko koneen arpoma luku suurempi, kuin syötetty (B%>A%) ja jos on, tuotetaan virhe tahallisesti (käyttäen ERROR-funktiota), jotta päästäisiin virherutiiniin. Jos taas syötetty luku on suurempi, kuin koneen arpoma, tuotetaan toinen virhe (ERROR 201) ja siirrytään näin virherutiiniin. Jos syötetty luku täsmää koneen arpoman luvun kanssa, tulostuu rivillä 70 tästä ilmoitus ja ohjelma päättyy riville 80 END-käskyyn. Rivillä 90 on REM-lause. Riviltä 100 alkaa virherutiini, jolla testataan virheen numeroa, joka virheen on aiheuttanut. Jos virheen virhekoodi oli 200, tulostuu teksti "LIIAN PIENI...". Jos virheen koodi ei ole 200, täytyy sen olla 201 ja siksi rivillä 100 oleva jälkimmäinen PRINT-lause. Rivillä 110 palataan virherutiinista RESUME-käskyllä.

ON INTERVAL GOSUB

MUOTO: ON INTERVAL = <X> GOSUB <nro>

VERSIOT: MSX: KASET. + LEVYKE
SVI: KASET. + LEVYKE

TARKOITUS: Funktiolla määritellään rivinumero, jolle hypätään jokaisen ajanjakson <X> välein.

LISÄYKSET: <nro> = rivinumero, jolta keskeytyksen jälkeen suoritus jatkuu.

<X> = kokonaisluku väliltä 1-65535, jolla määritellään keskeytysten aikaväli. Aikaväli voidaan laskea sekunneissa seuraavan kaavan mukaisesti: X/50 =sekunnit. Tarkimmillaan keskeytys saadaan siis 50 kertaa sekunnissa, jos ilmaisulle X annetaan arvo yksi (1).
Jos ilmaisu X määritellään nollaksi (0), tulostuu virheilmoitus "Illegal function call".
Siirryttäessä keskeytyspalvelurutiiniin (GOSUB-aliohjelma) suoritetaan automaattinen INTERVAL STOP -funktio, jolla estetään muut keskeytykset. RETURN-paluukäsky aliohjelman lopussa suorittaa automaattisen INTERVAL ON -funktion, jolla sallitaan keskeytykset. Jos aliohjelmassa on keskeytykset kieltävä INTERVAL OFF -käsky, kumoutuu automaattisesti INTERVAL ON -funktio.
ON ERROR -keskeytys kumoaa kaikki muut keskeytykset (ERROR, INTERVAL, KEY, STRIG, STOP ja SPRITE) eli sillä on korkein prioriteetti.
INTERVAL-keskeytykset on sallittava ennen 'ON INTERVAL GOSUB' -lausetta 'INTERVAL ON' -lauseella.
Funktio EI toimi komentotilassa.

ESIM:

```
10  INTERVAL ON
20  ON INTERVAL=50 GOSUB 100
30  PRINT "KESKEYTYSPALVELU HARJOITUS"
40  PRINT " OHJELMA ANTAA PIIP-ÄÄNEN"
50  PRINT "JOKA SEKUNTI. OHJELMAN VOI"
60  PRINT " PYSÄYTTÄÄ PAINAMALLA:"
70  PRINT "          CTRL+STOP"
80  GOTO 80
90  REM seuraavaksi keskeytysaliohjelma
100 BEEP
110 RETURN 30
```

Rivillä 10 sallitaan sisäiset keskeytykset. Rivillä 20 määritellään keskeytysväliksi yksi sekunti (50/50= 1 sek). Rivit 30—70 tulostavat otsikon. Rivillä 80 tehdään ikuinen ohjelmasilmukka GOTO-käskyllä, jonka aikana odotetaan seuraavaa keskeytystä. Rivillä 90 on REM-lause. Riviltä 100 alkaa keskeytysaliohjelma, jolla tuotetaan "piip"-ääni. Riviltä 110 suoritetaan paluu pääohjelmaan riville 30, jolta alkaa tulostus uudestaan.

ON KEY GOSUB

MUOTO: **ON KEY <rivi1> [,<rivi2>] [,<rivi3>, ...]**

VERSIOT: **MSX: KASET. + LEVYKE**
SVI: KASET. + LEVYKE

TARKOITUS: Funktiolla määritellään rivinumerot, joille hypätään painettaessa vastaavaa funktionäppäintä.

LISÄYKSET: <rivi1> = aliohjelman ensimmäinen rivi, jolle suoritus siirtyy painettaessa funktionäppäintä numero yksi (1).
Funktio toimii kuten 'ON <lause> GOSUB <nro> ' -lausekin paitsi nyt aliohjelmaan siirtymiseen vaikuttavat vain funktionäppäimet.
Siirryttäessä keskeytyspalvelurutiiniin (GOSUB-aliohjelmaan) suoritetaan automaattinen KEY STOP -funktio, jolla estetään muut keskeytykset. RETURN-paluukäsky aliohjelman lopussa suorittaa automaattisen KEY ON -funktion, jolla sallitaan keskeytykset. Jos aliohjelmassa on keskeytykset kieltävä KEY OFF -käsky, kumoutuu automaattinen INTERVAL ON -funktio.
ON ERROR -keskeytys kumoaa kaikki muut keskeytykset (ERROR, INTERVAL, KEY, STRIG, STOP ja SPRITE) eli sillä on korkein prioriteetti.
KEY-keskeytykset on sallittava ennen 'ON KEY GOSUB' -lausetta 'KEY ON' -lauseella.
Funktio EI toimi komentotilassa.

ESIM:

```
10  KEY(1) ON : KEY(2) ON
20  PRINT "PAINA FUNKTIONÄPPÄINTÄ 1 TAI 2"
30  ON KEY GOSUB 50, 80
40  GOTO 40
50  PRINT
60  PRINT "PAINOIT FUNKTIONÄPPÄINTÄ 1"
70  RETURN 20
80  PRINT
90  PRINT "PAINOIT FUNKTIONÄPPÄINTÄ 2"
100 RETURN 20
```

Rivillä 10 sallitaan funktionäppäinten yksi (1) ja kaksi (2) aiheuttamat keskeytykset. Rivillä 20 tulostetaan ohjeet. Rivillä 30 määritellään seuraavat rivit, joille hypätään painettaessa funktionäppäimiä yksi (rivi 50) tai kaksi (rivi 80) (HUOM! funktionäppäimet 3—10 eivät vaikuta). Rivillä 40 tehdään ikuinen silmukka GOTO-käskyllä. Rivit 50 ja 70 tulostavat yhden tyhjän rivin. Rivit 60 ja 90 tekstiä. Riveillä 70 ja 100 suoritetaan paluu aliohjelmasta RETURN-käskyllä.

ON STRIG GOSUB

MUOTO: **ON STRIG GOSUB <nro1>[,<nro2>] [,<nro3>]**

VERSIOT: **MSX: KASET. + LEVYKE**
SVI: KASET. + LEVYKE

TARKOITUS: Funktiolla määritellään rivinumerot, joilta suoritus jatkuu painettaessa peliohjaimen "Fire"-näppäintä (tai näppäimistöltä välilyöntiä).

LISÄYKSET: <nro1> = rivinumero, jolle siirrytään painettaessa näppäimistöltä välilyöntiä (spacebar).
<nro2> = rivinumero, jolle siirrytään painettaessa peliohjaimen yksi (1) tulitusnäppäintä (Fire).
<nro3> = rivinumero, jolle siirrytään painettaessa peliohjaimen kaksi (2) tulitusnäppäintä.
Siirryttäessä keskeytyspalvelurutiiniin (GOSUB-aliohjelmaan) suoritetaan automaattinen 'STRIG STOP' -funktio, jolla estetään muut keskeytykset. RETURN-paluukäsky aliohjelman lopussa suorittaa automaattisen 'STRIG ON' -funktion, jolla sallitaan keskeytykset. Jos aliohjelmassa on keskeytykset kieltävä 'STRIG OFF' -käsky, kumoutuu automaattinen 'STRIG ON' -funktio.
'ON ERROR' -keskeytys kumoaa kaikki muut keskeytykset (ERROR, INTERVAL, KEY, STRIG, STOP ja SPRITE) eli sillä on korkein prioriteetti.
STRIG-keskeytykset on sallittava ennen 'ON STRIG GOSUB' -lausetta 'STRIG (<nro>) ON' -lauseella.
Funktio EI toimi komentotilassa.

ESIM:

```
10  STRIG(0) ON
20  ON STRIG GOSUB 50
```

```
30  PRINT "OLEMME GOTO-SILMUKASSA ..."
40  GOTO 40
50  PRINT "PAINOIT VÄLINÄPPÄINTÄ !"
60  BEEP
70  RETURN 30
```

Rivillä 10 sallitaan välilyöntinäppäimestä aiheutuvat keskeytykset ja rivillä 20 määritellään rivi, jolle hypätään painettaessa välilyöntiä. Rivillä 30 tulostetaan tekstiä. Rivin 40 'GOTO 40' -lause muodostaa ikuisen silmukan. Riville 50 tullaan välilyönnin painalluksen jälkeen, jolloin tulostetaan teksti ja seuraavalla rivillä annetaan "piip" -ääni. Rivillä 70 suoritetaan paluu takaisin pääohjelmaan.

ON STOP GOSUB

MUOTO: **ON STOP GOSUB <nro>**

VERSIOT: **MSX: KASET. + LEVYKE**
SVI: KASET. + LEVYKE

TARKOITUS: Funktiolla määritellään rivi, jolle hypätään painettaessa CTRL- ja STOP-näppäimiä yhtaikaa.

LISÄYKSET: <nro> = rivi, jolta aliohjelma alkaa. Voi olla kokonaisluku väliltä 0-65529.
Siirryttäessä keskeytyspalvelurutiiniin (GOSUB-aliohjelmaan) suoritetaan automaattinen 'STOP STOP' -funktio, jolla estetään muut keskeytykset. RETURN-paluukäsky aliohjelman lopussa suorittaa automaattisen 'STOP ON' -funktion, jolla sallitaan keskeytykset. Jos aliohjelmassa on keskeytykset kieltävä 'STOP OFF' -käsky, kumoutuu automaattinen 'STOP ON' -funktio.
'ON ERROR' -keskeytys kumoaa kaikki muut keskeytykset (ERROR, INTERVAL, KEY, STRIG, STOP ja SPRITE) eli sillä on korkein prioriteetti.
STOP-keskeytykset on sallittava ennen 'ON STOP GOSUB' -lausetta 'STOP ON' -lauseella.
Funktio EI toimi komentotilassa.

ESIM:

```
10  STOP ON
20  ON STOP GOSUB 50
30  PRINT "OLEMME GOTO-SILMUKASSA ..."
40  GOTO 40
50  PRINT "PAINOIT CTRL+STOP !"
60  BEEP
70  RETURN 30
```

Rivillä 10 sallitaan näppäinten CTRL+STOP aiheuttama keskeytys (=Break). Rivillä 20 määritellään rivi, jolle hypätään annettaessa "Break". Rivillä 30 tulostetaan tekstiä. Rivin 40 'GOTO 40' -lause muodostaa ikuisen silmukan. Riviltä 50 alkaa aliohjelma, jolle tullaan keskeytyksen tapahduttua. Rivillä 60 tuotetaan "piip"-ääni. Riviltä 70 suoritetaan paluu pääohjelmaan riville 30. Ohjelmaa ei saa pysähtymään millään, sillä ainoa keino ohjelman pysäyttämiseksi olisi painaa CTRL+STOP, jonka vaikutus kuitenkin on estetty tällä kertaa ohjelmallisesti. Kun haluat pysäyttää ohjelman, käännä POWER-kytkin hetkeksi OFF-asentoon. Näin voit estää ohjelman listauksen ohjelmallisesti.

ON SPRITE GOSUB

MUOTO: **ON SPRITE GOSUB <nro>**

VERSIOT: **MSX: KASET. + LEVYKE**
SVI: KASET. + LEVYKE

TARKOITUS: Funktiolla määritellään rivi, jolle hypätään kahden tai useamman SPRITE-grafiikka-alkion kohdatessa.

LISÄYKSET: <nro> = rivi, jolle suoritus siirtyy alkioiden kohdatessa.
Siirryttäessä keskeytyspalvelurutiineihin (GOSUB-aliohjelmaan) suoritetaan automaattinen 'SRPITE STOP' -funktio, jolla estetään muut keskeytykset. RETURN-paluukäsky aliohjelman lopussa suorittaa automaattisen 'SPRITE ON' -funktion, jolla sallitaan keskeytykset. Jos aliohjelmassa on keskeytykset kieltävä 'SPRITE OFF' -käsky, kumoutuu automaattinen 'SPRITE ON' -funktio.
'ON ERROR' -keskeytys kumoaa kaikki muut keskeytykset (ERROR, INTERVAL, KEY, STRIG, STOP ja SPRITE) eli sillä on korkein prioriteetti.

SPRITE-keskeytys on sallittava ennen 'ON SPRITE GOSUB' -lausetta 'STOP ON' -lauseella.
Funktio EI toimi komentotilassa. Ja ainoastaan 'SCREEN 2' - ja 'SCREEN 3' -moodeissa (MSX-BASIC) tai 'SCREEN 1' - ja 'SCREEN 2' -moodeissa (SVI-BASIC).

ESIM:

```
10  SPRITE ON
20  ON SPRITE GOSUB 220
30  SCREEN 3,2
40  FOR A%=1 TO 32
50  READ B%
60  SPRITE$=SPRITE$+CHR$ (B%)
70  NEXT
80  SPRITE$(0)=SPRITE$
90  PUT SPRITE 0,(35,85),8,0
100 GOSUB 210
110 PUT SPRITE 1,(110,85),11,0
120 GOSUB 210
130 PUT SPRITE 2,(185,85),8,0
140 GOSUB 210
150 PUT SPRITE 3,(115,85),15,0
160 REM seuraavaksi datat sprite-alkioihin
170 DATA 3,15,31,57,121,255,255,255
180 DATA 42,42,42,74,74,82,146,146
190 DATA 192,240,248,156,158,255,255,255
200 DATA 84,84,84,82,82,74,73,73
210 FOR A=1 TO 1000 : NEXT : RETURN
220 FOR A=1 TO 1500 : NEXT : BEEP : STOP
```

Rivillä 10 sallitaan SPRITE-kuva-alkioiden kohtaamisesta aiheutuva keskeytys ja rivillä 20 määritellään rivi, jolle alkioiden kohdatessa siirrytään. Rivillä 30 annetaan käsky siirtyä tarkkuusgrafiikkaan (HUOM! SPECTRAVIDEO:lla tulee rivi 30 kirjoittaa seuraavasti: 30 SCREEN 2,3). Rivit 40—70 muodostavat FOR-NEXT -silmukan, jonka aikana muodostetaan SPRITE-alkio. Rivin 50 READ-käsky lukee luvun DATA-lauseilta (ensimmäinen data on rivin 170 ensimmäinen luku, luku 3). Rivillä 60 muunnetaan luku SPRITE-alkion yhdeksi osaksi CHR$-funktion avulla. Rivin 70 NEXT-käsky sulkee silmukan. Kun kaikki 32 dataa on luettu (SPRITE-alkion muoto on määritelty), määritellään luettu SPRITE-hahmo SPRITE-alkioksi numero nolla (0). Rivillä 90 tulostetaan ensimmäinen SPRITE-alkio (kts. PUT SPRITE -käsky) ja rivillä 100 hypätään aliohjelmaan kuluttamaan aikaa (jotta ohjelman kulkua voitaisiin seurata, sillä muuten kaikki tapahtuisi niin nopeasti, että oleellinen jäisi huomaamatta). Rivillä 110 tulostuu sama SPRITE-alkio, tosin eri värillä, keskelle kuvaruutua ja rivillä 120 hypätään taas kuluttamaan aikaa. Rivillä 130 tulostuu seuraava SPRITE-alkio ja rivillä 140 hypätään taas looppiin. Rivillä 150 tulostuu SPRITE-alkio osittain keskimmäisen SPRITE-alkion päälle, jolloin aiheutuu keskeytys. Silloin hypätään riville 220 aliohjelmaan. Rivillä 160 on REM-lause ja rivit 170—200 sisältää SPRITE-alkion datat. Rivillä 210 on vain aikaa kuluttava FOR-NEXT -silmukka. Rivillä 220 on aliohjelma, jolle suoritus siirtyy SPRITE-alkioiden kohdatessa. Ensin kulutetaan vain aikaa, jotta SPRITE-hahmojen "törmäys" voitaisiin havaita ja sitten tuotetaan "piip" -ääni, keskeytetään ohjelma STOP-käskyyn ja palataan komentotilaan.

INTERVAL ON/OFF/STOP

MUOTO:

```
INTERVAL ON
INTERVAL OFF
INTERVAL STOP
```

VERSIOT:

MSX: KASET. + LEVYKE
SVI: KASET. + LEVYKE

TARKOITUS: Funktiolla sallitaan, kielletään tai estetään sisäisen ajastimen aiheuttamat keskeytykset.

LISÄYKSET: 'INTERVAL ON' -lauseella sallitaan sisäisen ajastimen keskeytykset.
Jos annetaan 'INTERVAL OFF' -käsky, estyy keskeytykset ja kaikki INTERVAL-keskeytykset käskyn INTERVAL ON antamiseen asti unohdetaan.
Jos annetaan 'INTERVAL STOP' -käsky, estyy keskeytykset, mutta jos keskeytyksiä kuitenkin tulee, taltioidaan ne muistiin ja heti 'INTERVAL ON' -käskyn jälkeen hypätään 'ON INTERVAL GOSUB' -lauseella määritellylle riville.

ESIM:

```
10  INTERVAL ON
20  ON INTERVAL=5 GOSUB 40
```

```
30  GOTO 30
40  A%=A%+1
50  PRINT TAB (16); A% : PRINT CHR$(30);
60  A$=INKEY$ : IF A$="" " THEN RETURN
70  INTERVAL STOP
80  A$=INKEY$ : IF A$="" " THEN 80
90  GOTO 10
```

Rivillä 10 sallitaan keskeytykset ja rivillä 20 määritellään keskeytysväli (5/50 = 1/10 osa sekunti) ja rivi, jolle hypätään keskeytyksen tapahduttua. Rivin 30 GOTO-käsky muodostaa silmukan, jossa ohjelma odottaa keskeytyksiä. Rivillä 40 kasvatetaan kokonaislukulaskuria yhdellä. Rivillä 50 tulostetaan laskurin arvo sarakkeeseen 16 ja siirrytään yksi rivi ylöspäin (CHR$(30) -funktio). Rivillä 60 tunnustellaan, painetaanko mitään näppäintä ja jos ei, palataan takaisin pääohjelmaan (riville 30). Jos jotain näppäintä painetaan, siirrytään riville 70, jolla estetään keskeytykset toistaiseksi. Rivillä 80 tunnustellaan taas näppäimistöä ja jos jotain näppäintä painetaan, siirtyy suoritus riville 90, jolla palataan ohjelman alkuun.

KEY ON/OFF/STOP

MUOTO:

KEY (<nro>) ON
KEY (<nro> OFF
KEY (<nro>) STOP

VERSIOT:

MSX: KASET. + LEVYKE
SVI: KASET. + LEVYKE

TARKOITUS:

Funktiolla sallitaan, kielletään tai estetään funktionäppäinten aiheuttamat keskeytykset.

LISÄYKSET:

KÄSKY		VAIKUTUS
KEY (<nro>) ON	;	sallitaan funktionäppäimen <nro>) keskeyttävä vaikutus.
KEY (<nro>) OFF	;	kielletään funktionäppäimen <nro> keskeyttävä vaikutus.
KEY (<nro>) STOP	;	estetään toistaiseksi funktionäppäimen <nro> keskeyttävä vaikutus.
KEY ON	;	SVI-BASIC: sallitaan kaikkien funktionäppäinten keskeyttävä vaikutus. MSX-BASIC: asetetaan funktionäppäinten merkitykset näkyviin (ei liity keskeytyksiin).
KEY OFF	;	SVI-BASIC: kielletään kaikkien funktionäppäinten keskeyttävä vaikutus. MSX-BASIC: poistetaan funktionäppäinten merkitykset kuvaruudulta (ei liity keskeytyksiin).
KEY STOP	;	SVI-BASIC: muiden, paitsi viimeksi keskeyttäneen funktionäppäimen keskeyttävä vaikutus estetään toistaiseksi. MSX-BASIC: ei käytössä.

Ilmaisu <nro> on funktionäppäintä vastaava kokonaisluku tai lause väliltä 1—10.

ESIM:

```
10  KEY(1) ON : KEY(2) ON
20  KEY(3) ON : KEY(4) ON
30  KEY(5) ON
40  ON KEY GOSUB 70,90,110,140,160
50  PRINT "PAINA JOTAIN FUNKTIONÄPPÄINTÄ 1—5"
60  GOTO 60
70  PRINT "PAINOIT F.NÄPPÄINTÄ 1"
80  RETURN 50
90  PRINT "PAINOIT F.NÄPPÄINTÄ 2"
100 RETURN 50
110 PRINT "PAINOIT F.NÄPPÄINTÄ 3, MUTTA KOETA UUDESTAAN..."
120 KEY(3) OFF
130 RETURN 50
140 PRINT "PAINOIT F.NÄPPÄINTÄ 4"
150 RETURN 50
160 PRINT "PAINOIT F.NÄPPÄINTÄ 5"
170 PRINT
180 PRINT "        END"
190 END
```

Riveillä 10—30 sallitaan funktionäppäinten 1—5 keskeytykset ja rivillä 40 määritellään rivit, joille hypätään kutakin näppäintä painettaessa. Rivi 50 tulostaa ohjeet. Rivillä 60 on GOTO-silmukka. Rivillä 70 tulostuu teksti f.n. numero yksi painalluksesta. Rivillä 80 suoritetaan paluu riville 50. Rivin 90 PRINT-lause tulostaa vastaavasti f.n. numero kaksi painalluksesta. Rivillä 100 paluu riville 50. Rivillä 110 tulostetaan teksti f.n. kolme painalluksesta. Rivillä 120 kielletään f.n. numero kolme keskeytykset jatkossa. Rivillä 130 paluu riville 50. Rivit 140 ja 160 tulostavat vastaavan f.n. painalluksesta. Rivin 170 PRINT-käsky tulostaa tyhjän rivin. Ohjelman saa loppumaan painamalla funktionäppäintä numero viisi (5).

STRIG ON/OFF/STOP

MUOTO:
STRIG(<nro>) ON
STRIG(<nro>) OFF
STRIG(<nro>) STOP

VERSIOT:
MSX: KASET. + LEVYKE
SVI: KASET. + LEVYKE

TARKOITUS: Funktiolla sallitaan, kielletään tai estetään välinäppäimen tai peliohjaimen "Fire"-näppäimen aiheuttamat keskeytykset.

LISÄYKSET: Jos ilmaisu <nro> on 0, tarkoitetaan välilyöntinäppäintä, jos <nro> on 1, tarkoitetaan peliohjaimen yksi tulitusnäppäintä ja jos <nro> on 2, tarkoitetaan peliohjaimen kaksi tulitusnäppäintä ("Fire"-näppäin).

ESIM:

KÄSKY		VAIKUTUS
STRING(0) ON	;	sallii välilyöntinäppäimen aiheuttamat keskeytykset.
STRING(1) OFF	;	kieltää peliohjaimen yksi (1) aiheuttamat keskeytykset.
STRING(1) STOP	;	estää toistaiseksi peliohjaimen yksi (1) keskeytykset.

STOP ON/OFF/STOP

MUOTO:
STOP ON
STOP OFF
STOP STOP

VERSIOT:
MSX: KASET. + LEVYKE
SVI: KASET. + LEVYKE

TARKOITUS: Funktiolla sallitaan, kielletään tai estetään näppäinten CTRL ja STOP aiheuttama keskeytys (=Break).

LISÄYKSET:

KÄSKY		VAIKUTUS
STOP ON	;	käsky sallii aliohjelmaan siirtymisen painettaessa CTRL- ja STOP-näppäimiä.
STOP OFF	;	käsky kieltää aliohjelmaan siirtymisen painettaessa CTRL- ja STOP-näppäimiä.
STOP STOP	;	käsky estää toistaiseksi 'ON STOP GOSUB' -lauseella määriteltyyn aliohjelmaan siirtymisen, mutta jättää keskeytyksen sattuessa tiedon siitä muistiin ja seuraavan 'STOP ON' -lauseen jälkeen hypätään heti aliohjelmaan.

ESIM:

```
10  ON STOP GOSUB 50
20  STOP ON
30  PRINT "PAINA CTRL+STOP"
40  GOTO 40
50  PRINT " KIITOS !"
60  RETURN
```

Rivillä 10 määritellään rivi, jolle suoritus siirtyy painettaessa näppäimiä CTRL ja STOP yhtaikaa. Rivillä 20 sallitaan kyseisten näppäinten aiheuttama keskeytys (siirtyminen aliohjelmaan). Rivillä 30 tulostetaan ohjeita. Rivin 40 GOTO-silmukka odottaa CTRL+STOP:n painamista. Rivillä 50 tulostetaan tekstiä ja aliohjelman toisella rivillä suoritetaan paluu pääohjelmaan. Huom! ohjelman saa pysähtymään vain kääntämällä POWER-kytkimen hetkeksi OFF-asentoon.

SPRITE ON/OFF/STOP

MUOTO: **SPRITE ON**
SPRITE OFF
SPRITE STOP

VERSIOT: **MSX: KASET. + LEVYKE**
SVI: KASET. + LEVYKE

TARKOITUS: Funktiolla sallitaan, kielletään tai estetään kahden tai useamman SPRITE-alkion kohtaamisesta aiheutuva keskeytys.

LISÄYKSET: SPRITE-grafiikassa voidaan tutkia kahden SPRITE-alkion kohtaamista ja suorittaa sen mukaisia toimenpiteitä. Jos kohtaamisen halutaan aiheuttavan keskeytyksen, annetaan ohjelman alussa tai ennen mahdollista kohtaamista 'SPRITE ON' -käsky. Kohtaavien SPRITE-hahmojen numeroa ei voida BASIC:llä testata, ilman hankalaa taulukointia. Kts. 'ON SPRITE GOSUB'-funktio.

ESIM:

KÄSKY		**VAIKUTUS**
SPRITE ON	;	käsky sallii kahden tai useamman SPRITE-hahmon kohdatessa hypyn 'ON SPRITE GOSUB'-lauseella määriteltyyn aliohjelmaan.
SPRITE OFF	;	käsky kieltää aliohjelman siirtymisen hahmojen kohdatessa.
SPRITE STOP	;	käsky estää toistaiseksi aliohjelmaan hypyn, vaikkakin hahmot kohtaavat.

MUSIIKKIKÄSKYT

MSX-kotitietokoneissa ja SPECTRAVIDEO:ssa on kolme erillistä ohjelmoitavaa äänigeneraattoria, jotka ovat yhdellä mikroprosessoripiirillä. Äänigeneraattorit voidaan ohjelmoida tuottamaan kukin omaa säveltä täysin toisistaan riippumatta. Äänien ja musiikin tuottamiseen on varattu käskyt PLAY ja SOUND (BEEP-funktio on selvitetty muussa yhteydessä).

PLAY

MUOTO: **PLAY [<string1>] [,< string2>] [,<string3>]**

VERSIOT: **MSX: KASET. + LEVYKE**
SVI: KASET. + LEVYKE

TARKOITUS: Käskyllä ohjataan äänisyntetisaattoreita soittamaan halutut nuotit.

LISÄYKSET: Käskyä voidaan käyttää komentotilassa tai ohjelmallisesti. '
Komentotilassa PLAY-käsky annetaan seuraavasti:

PLAY "EEFAD"

Tämä esimerkki soittaa nuotit E, E, F, A ja D. Nuotit annetaan siis erotettuina lainausmerkeillä. Nuotit voidaan kirjoittaa joko isoilla tai pienillä kirjaimilla.
Ohjelmallisesti PLAY-käsky toteutetaan seuraavasti:

```
10  PLAY "ADEFG"
20  PLAY "Cdefgacc"
RUN <ENTER>
(... äänisyntetisaattori toistaa nuotit...)
Ok
```

Esimerkissä rivillä 10 soitetaan viisi säveltä, jonka jälkeen siirrytään riville 20 soittamaan nuotit C, D, E,... .
Soitettavat nuotit voidaan sijoittaa myös merkkijonon sisällöksi. Tällöin PLAY-käsky käsittelee vastaavaa merkkijonoa ryhmänä säveliä.

```
10  A$ = "CDFE"
20  PLAY A$
```

Tässä esimerkissä määritellään merkkijonon A sisällöksi nuotit C, D, F ja E. Rivillä 20 soitetaan vastaavat nuotit lukien merkkijonon A sisältöä.

PLAY-käskyn käytettävissä on seuraavat sävelet:

A, B, C, D, E, F, G ja ylänuottia vastaavat merkinnät # tai + sekä alanuottia vastaava merkintä -.

Jotta ylä- tai alanuottia vastaavaa merkintää voitaisiin käyttää, tulee soitettavan sävelen vastata pianon mustaa kosketinta. Siksi esim. PLAY "B+" on laiton käsky.
Soitettavien sävelien väliin voidaan lisätä ohjainkomentoja, joilla vaikutetaan oktaaviin, sävelen kestoon, jne. (=musiikin makrokieli). Kaikilla ohjaintavuilla on omat oletusarvonsa, jotka palautuvat, jos PLAY-funktiossa on kielioppi (Syntaksi) virhe. Määritellyn ohjaustavun vaikutus kestää seuraavaan saman ohjaustavun uuteen määrittelyyn asti (tai virheeseen, jolloin palautuu oletusarvo). Ohjaintavut annetaan aina ennen nuottia ja ohjaintavu voi olla jokin seuraavista:

V (Volume eli äänenvoimakkuus)
MUOTO: V<nro>

Ohjaintavulla määritellään toistettavan nuotin/nuottien äänenvoimakkuus. Ilmaisu <nro> voi saada arvot: 0—15 (0=hiljaa, 15=kovaa). Oletusarvo on kahdeksan (8).

ESIM: **PLAY "v1ADv3ADv5ADv7ADv9ADv11ADv13ADv15AD" <ENTER>**

Äänenvoimakkuutta kasvatetaan kerta kerralta koko ajan kahden yksikön verran kerralla. Jokaisen voimakkuuden määrittelyn jälkeen soitetaan nuotit A ja D. Alkuperäisen äänenvoimakkuuden voit palauttaa seuraavasti: PLAY "v8". Ilmaisu V on merkitty selvyyden vuoksi pienellä kirjaimella (v).

O (Octave eli oktaavi)
Muoto: O<nro>

Tavulla määritellään soitettavan sävelen oktaavi. Oktaavi <nro> voi olla välillä 1—8. Oletusarvona neljä (4).

ESIM: **PLAY "o1ADo2ADo3ADo4ADo5ADo6ADo7ADo8AD" <ENTER>**

Oktaavit esitellään alkaen alimmasta ylimpään. Jokaisen oktaavin kohdalla soitetaan sävelet A ja D.

T (Tempo eli rytmin nopeus)
MUOTO: T<nro>

Tavulla voidaan vaihtaa sävelmän rytmiä. Tempo ilmaistaan arvoilla 32—255 (32=nopea, 255=hidas). Oletusarvona 120.

ESIM: **PLAY "t32ADCEt255ADCE" <ENTER>**

Esimerkki soittaa sävelet A, D, C ja E ensin nopeasti ja sitten hitaammalla mahdollisella rytmillä.

L (Length eli nuotin pituus, kestoaika)
MUOTO: L<nro>

Ohjaintavulla voidaan jokaisen nuotin kestoaika eli pituus määritellä erikseen, jolloin vältytään useamman nuotin peräkkäiseltä kirjoitukselta ja sävelen kesto saadaan tarkalleen määriteltyä. Kestoaika voi olla välillä 1—64 (nuotin todellinen kestoaika saadaan jakamalla luku neljä (4) ilmaisun <nro> arvolla). Oletusarvo on neljä (4).

ESIM:

L		KESTOAIKA
1	;	neljä nuottia.
2	;	kaksi nuottia.
3	;	4/3-osa nuottia.
...		...
64	;	16:s osa nuottia.

Nuotin kestoaika voidaan antaa myös ilman kirjainta "L", jolloin merkintä on seuraava:

PLAY "A32D42" ja PLAY "L32AL42D" vastaavat toisiaan.

S (Shape eli nuotin "muoto")
MUOTO: S<nro>

Ohjaintavulla vaikutetaan nuotin muotoon, joita on valittavana kahdeksan (8) erilaista. Ilmaisu <nro> voi saada arvot 1—15, oletusarvona 1. Valittavat nuottien muodot ovat seuraavat:

ESIM:

S	MUOTO
0,1,2,3,9	
4,5,6,7,15	

8

10
11

12

13

14

ESIM: **PLAY "s10ADFEEC" <ENTER>**

Käsky soittaa nuotit A, D, F, E, E ja C muodolla 10.

M (Modulation eli muodon jaksollisuus)
MUOTO: M<nro>

Ohjaintavulla vaikutetaan jaksollisuuteen, jolla säveltä toistetaan voimassa olevalla muodolla. Ilmaisu <nro> voi saada arvot 1—65535 ja oletusarvona on 255.

ESIM: **PLAY "M2000ADCEM10000ADCE" <ENTER>**

Käsky soittaa nuotit A, D, C ja E kahdella eri jaksolla.

R (Rest eli tauko)
MUOTO: R<nro>

Ohjaintavulla R + <nro> saadaan aikaan tauko kahden nuotin välille, jonka pituus määräytyy arvon <nro> mukaan. <nro> voi saada arvot 1—64, oletusarvona neljä (4). Arvoa yksi vastaa pitkä tauko ja arvoa 64 lyhyt tauko.

. (piste)

Piste nuotin jäljessä aiheuttaa nuotin kestoajan muunnon sen hetkisestä pituudesta luvulla 3/2 kerrottuun pituuteen. Pisteitä voidaan sijoittaa myös useita peräkkäin, jolloin vastaava kestoaika lasketaan sen mukaisesti. Esimerkiksi kolme (3) pistettä nuotin jäljessä saa nuotin kestoksi 27/8 kertaa sen hetkinen arvo. Pisteitä voidaan käyttää myös R (Rest) ohjaintavun jälkeen, jolloin tauon pituus määräytyy edellä esitetyn kaavan mukaisesti.

ESIM: **PLAY "AECR1AECR21AECR42AECR64AEC" <ENTER>**

Käsky toistaa nuotit A, E ja C peräkkäin, vaihdellen nuottien välistä taukoa suurimmasta pienimpään.

N (Note eli nuotti)
MUOTO: N<nro>

Ohjaintavulla soitetaan haluttu nuotti <nro>, jossa <nro> voi saada arvot 0—96. Arvo nolla (0) vastaa taukoa. Funktio käsittelee lukuja nuotteina, alkaen oktaavista 0 oktaaviin 7 (kertaa kaksitoista säveltä = 96 eri säveltä).

ESIM: **PLAY "N36" vastaa käskyä PLAY "o4C".**

USEAMPI ÄÄNINEN TOISTO

Esimerkit tähän mennessä ovat käsittäneet vain yksiäänistä toistoa. PLAY-käsky sallii jopa kolmiäänisen toiston samanaikaisesti. Jokainen ääni voidaan ohjelmoida tuottamaan säveliä omalla oktaavilla, tempolla, sävelien kestoajoilla jne. Jokaisen äänen sävelet kirjoitetaan lainausmerkkien sisään (poikkeuksena merkkijono-ilmaisut) ja erotetaan toisistaan pilkulla.

ESIM:
```
10  A$="o3L40ADEFC"
20  B$="o5S10M4000GGFFA"
30  PLAY A$,"o4FDEAC",B$
```

Rivillä 10 määritellään muuttujaan A$ kuuluviksi nuotit A, D, E, F ja C, jotka soitetaan oktaavissa kolme, pituudella 40. Rivillä 20 määritellään vastaavasti merkkijonon B$ nuotit omassa oktaavissa ja muodossa. Rivillä 30 annetaan PLAY-käsky, joka soittaa kolmiäänisesti mielivaltaista sävelmää (huomaa, että keskimmäisen äänen sävelet on määritelty vasta PLAY-käskyn yhteydessä, eikä aikaisemmin, kuten kaksi muuta ääntä).

SOUND

MUOTO: SOUND <rek>,<X>

VERSIOT: MSX: KASET. + LEVYKE
SVI: KASET. + LEVYKE

TARKOITUS: Funktiolla kirjoitetaan PSG:n (Programmable Sound Generator) rekisteriin <rek> tieto <X>.

LISÄYKSET: PSG-piiri sisältää 13 eri rekisteriä (rekisteri = yhden tavun eli 8:n bitin "tietolokero"), jotka on numeroitu 1—13.
Rekisterin sisältämä tieto on kokonaisluku väliltä 0—255. Jokaisella rekisterillä on oma merkityksensä, jotka seuraavaksi käsitellään. PLAY-käskyn käyttö musiikin ja efektien tuottamiseen tuntuu helpommalta ja onkin sitä, mutta SOUND-käsky mahdollistaa monia muita toimintoja.
Rekistereiden merkitykset ovat seuraavat:

REK.NUM	REKISTERI	TOIMINTA
0—5	sävelen ohjaus	tuottaa säveljaksoja.
6	kohinan ohjaus	saada aikaan kohinaa.
7	äänen sekoituksen ohjaus	rekisterillä mahdollistetaan sävel- ja/tai kohinakanavat.
8—10	äänen voimakkuuden ohjaus.	valitaan haluttu äänenvoimakkuus.
11—13	"sävel verhon" ohjaus.	valitaan haluttu muoto.

SÄVELEN OHJAUS REKISTERIT (0—5)

PSG:ssä on kolme ulostulo kanavaa: A, B ja C. Jokaiselle kanavalle valittava toistotaajuus voidaan laskea kaavasta:

$$\frac{3579545}{\text{haluttu taajuus} * 16} = X$$

Rekisterit A, B ja C ovat 16-bittisiä ja niille syötettävät arvot muodostetaan seuraavasti:
rekisterin alempi tavu = X AND 255
(X = tulos edellä esitetystä kaavasta, jonka kanssa suoritetaan AND-operaatio)
rekisterin ylempi tavu = X/256
(edellä esitetyn kaavan tulos jaetaan 256:lla)
Rekistereiden vastaavat numerot ovat:

KANAVA	REKISTERI	
A	1,0	(1 = ylempi tavu)
B	3,2	(3 = ---II---
C	5,4	(5 = ---II---

ESIM:

```
10  INPUT "ANNA TAAJUUS, JONKA HALUAT KUULLA"; A
20  T=3579545/(16*A)
30  H=T/256
40  L=T AND 255
50  SOUND 0,L
60  SOUND 1,H
70  SOUND 8,10
80  PRINT "SÄÄDÄN ÄÄNEN VOIMAKKUUTEEN 10."
90  PRINT "TOISTAN TAAJUUDEN KANAVASTA A."
100 SOUND 7,254
```

Rivillä 10 pyydetään käyttäjältä taajuutta, joka myöhemmin SOUND-käskyllä tuotetaan. Rivillä 20 lasketaan kanavan A rekistereille tarvittava ohjausarvo. Tarvittavat ohjaintavut lasketaan riveillä 30 ja 40 (H = ylempi tavu ja L = alempi tavu). Rivillä 50 kirjoitetaan A-kanavan sävelrekisterin alempaan tavuun haluttuun taajuuteen tarvittava ohjainluku ja rivillä 60 ylempään tavuun. Rivillä 70 määrätään äänenvoimakkuudeksi 10 (8 = A-kanavan äänenvoimakkuuden säädön rekisteri). Rivit 80 ja 90 tulostavat tekstiä. Rivillä 100 avataan äänikanava (saatetaan ääni kuuluviin).
Kuten huomaat, on äänen tuottaminen SOUND-käskyllä huomattavasti hankalampaa, kuin PLAY-käskyllä. SOUND-käskyn yhtenä etuna kuitenkin on, että ääniä voi tuottaa miltä tahansa taajuudelta.

ÄÄNENVOIMAKKUUDEN REKISTERIT (8-10)

Edellisessä esimerkissä käytimme äänenvoimakkuutta kymmenen (vrt. rivi 70). Äänenvoi-

makkuutta voimme ohjata siis ohjelmallisesti, eikä television/monitorin äänenvoimakkuuden säätöön tarvitse koskea. Äänenvoimakkuutta voidaan ohjata luvuilla 0—15, joista arvo 15 on voimakkain ja arvo nolla (0) vastaa hiljaisinta toistoa.

ESIM:

```
10  SOUND 0,100
20  SOUND 1,0
30  SOUND 7,254
40  FOR A% = 0 TO 15
50  SOUND 8,A%
60  FOR B% = 1 TO 100 : NEXT B%
70  NEXT A%
80  FOR A% = 15 TO 0 STEP -1
90  SOUND 8,A%
100 FOR B% = 1 TO 100 : NEXT B%
110 NEXT A%
120 GOTO 40
```

Riveillä 10 ja 20 annetaan A-kanavan taajuudensäätörekistereille ohjausarvot. Rivillä 30 avataan A-kanava. Rivit 40—70 muodostavat kaksi sisäistä FOR-NEXT -silmukkaa, joista uloimpi silmukka (riviltä 40 alkava) määrittelee äänenvoimakkuuden ja rivin 60 silmukka kuluttaa vain aikaa, jonka aikana toistetaan A-kanavan kautta ääntä. Rivin 70 NEXT-käsky sulkee rivin 40 FOR-silmukan. Rivit 80—110 muodostavat vastaavan silmukan, kuitenkin niin, että rivillä 80 määriteltävä äänenvoimakkuus (vrt. rivi 60) pienenee joka kierroksella. Rivin 120 GOTO-käsky tekee ohjelmasta ikuisen silmukan, jonka voi pysäyttää painamalla CTRL + STOP. Ohjelma toistaa äänen, joka muistuttaa sireeniä.
Äänenvoimakkuutta voidaan säätää myös ENVELOPE-rekisterin kautta (äänenlaajennusrekisteri). Tällöin asetetaan äänenvoimakkuus arvoon 16 ja säädetään voimakkuutta vastaavasti rekistereiden 11, 12 ja 13 avulla (11 = A-kanava, 12 = B-kanava ja 13 = C-kanava). Tästä esimerkki SOUND-käskyn viimeisessä kappaleessa.

ÄÄNENSEKOITUSREKISTERI (7)

Tämän rekisterin avulla vaikutetaan äänen sävyyn ja kohinaan. Jokaista kanavaa varten voidaan valita oma äänensävyn yhdistelmä. Tämän rekisterin (7) biteistä on käytössä vain kuusi alinta (bitit 0—5). Kaksi (2) ylintä on varattu I/O-linjoja varten.
Rekisterin bitit voivat olla siis joko nollia (0) tai ykkösiä (1). Nolla-bitillä mahdollistetaan (saatetaan ON-tilaan) ja ykkös-bitillä estetään (OFF-tila). Rekisterin biteillä on seuraavat merkitykset:

BITTI NUMERO	MERKITYS
0	; ääni kanavalle A.
1	; ääni kanavalle B.
2	; ääni kanavalle C.
3	; kohina kanavalle A.
4	; kohina kanavalle B.
5	; kohina kanavalle C.
6 ja 7	; EI BASIC:n käytössä.

ESIM:

```
KÄSKY                   ANTAA KÄSKYN
SOUND 7,254             ; avata kohinan kanavalle A (luku 254 = &B11111110).
SOUND 7, &B11011011     ; avata sekä äänen että kohinan kanavalle C.
```

ÄÄNEN LAAJUUDEN REKISTERIT (11,12)

Tämän rekisterin avulla tuotetaan monipuolisia ääniä/säveliä. Äänen muoto/stereovaikutelma voidaan valita joko rekistereillä 11 ja 12, tai käyttämällä rekisteriä 13 valitsemalla valmiina olevista aaltomuodoista tarkoitukseen sopiva.
Rekisteri on 16-bittinen (rekisterit 11 + 12) ja ohjelmoidaan siis kahdella SOUND-käskyllä. Haluttu äänenmuoto saadaan samalla kaavalla, kuin sävelohjausrekisterin äänimuoto (kts. SOUND-käskyn muotorekisteri).

ESIM:

```
10  SOUND 0,100
20  SOUND 1,0
30  SOUND 7,254
40  SOUND 8,16
50  SOUND 13,14
60  S = 0.5
70  KELLO = 3579545
80  L = KELLO/(256*S) AND 255
90  H = KELLO/(256*S)/256
100 SOUND 11,L
110 SOUND 12,H
```

Riveillä 10 ja 20 valitaan kanavan A taajuus ja rivillä 30 avataan A-kanava. Rivillä 40 mahdollistetaan äänenlaajuuden rekisterin käyttö ja rivillä 50 valitaan käytettävä aaltomuoto. Rivillä 60 määritellään muuttujalle S arvo .5, jota käytetään laskettaessa rekisterin alemman ja ylemmän tavun ohjausarvoja riveillä 80 ja 90. Riveillä 100 ja 110 kirjoitetaan saadut muodot rekistereihin 11 ja 12 (saatetaan haluttu ääni kuuluviin).

MUOTO REKISTERI (13)

Tällä rekisterillä valitaan käytettävä aaltomuoto. Muotoja on kahdeksan (8) erilaista ja ne valitaan neljällä alimmalla bitillä (bitit 0—3). Aaltomuodot ovat samat, kuin PLAY-käskyllä. Kertauksen vuoksi ne kuitenkin esitettäköötkin:

BITIT	AALTOMUOTO	VALITTU LUKU
3210		
00XX		0,1,2,3
01XX		4,5,6,7
1000		8
1001		9
1010		10
1011		11
1100		12
1101		13
1110		14
1111		15

Bitti "X" voi olla kumpi tahansa (nolla tai yksi).

ESIM:

```
10  SOUND 0,100
20  SOUND 1,0
30  SOUND 7,&B11111110
40  SOUND 8,16
50  SOUND 13,14
60  S=0.5
70  KELLO=3579545
80  L=KELLO/(256*S) AND 255
90  H=KELLO/(256*)/256
100 SOUND 11,L
110 SOUND 12,L
```

Riveillä 10 ja 20 valitaan taajuus. Rivillä 30 avataan kanava A ja rivillä 40 mahdollistetaan rekisterit 11 ja 12 (kts. edellinen kappale). Rivillä 50 valitaan käytettävä aaltomuoto. Rivillä 60 määritellään muuttujalle S arvoksi 0.5, jota käytetään laskettaessa ohjaintavuja riveillä 80 ja 90. Rivillä 70 määritellään muuttujan KELLO arvo. Riveillä 100 ja 110 kirjoitetaan saadut luvut rekistereihin ja näin saadaan ääni kuuluviin.

GRAFIIKKAKÄSKYT

Grafiikkakäskyjen osalta on MSX- ja SVI-BASIC:ssä eroavaisuuksia, mikä estää grafiikkaohjelmien hyödyntämisen ilman muutoksia. Muutokset, joita tarvitaan ohjelmien hyödyntämiseen näiden BASIC-murteiden välillä, ovat kuitenkin vähäisiä.
Jäljempänä esiintyvällä sanalla "grafiikkamoodi" tai "grafiikkamuoto" tarkoitetaan tilaa, jossa voidaan grafiikkakäskyjen avulla esittää mm. funktioiden kuvaajia, ympyröitä, viivoja, jne. Grafiikkamuotoon siirtyminen tapahtuu SCREEN-käskyllä, joka MSX- ja SVI-BASIC:ssä eroaa huomattavasti. Tästä syystä kaikissa jäljempänä olevissa esimerkkiohjelmissa on SCREEN-käsky MSX- ja SVI-BASIC:lle esitetty erikseen.
Grafiikkamoodia ei voida myöskään käyttää komentotilassa, sillä tulostus pysyy grafiikkamoodissa vain jonkin silmukan avulla (esim. käskyllä: 210 GOTO 210, jossa luku 210 on rivinumero). Grafiikkamuodosta poistutaan painamalla näppäimiä CTRL+STOP.
Sekä MSX- että SVI-BASIC:ssä on kuvaruutu grafiikkamoodissa jaettu 256* 192 pisteeseen. X-akselin suuntaisesti on pisteet numeroitu vasemmalta oikealle 0—255 (yht. 256 kpl). Y-akselin suunnassa on pisteet numeroitu ylhäältä alaspäin 0—191 (yht. 192 kpl). Kuvaruudun vasemman yläkulman (huomioi, että molemmat BASIC-murteet jättävät kuvaruudun ylä- ja alareunaan tyhjän tilan, johon emme grafiikkamoodissa voi tulostaa) koordinaatit ovat siis 0,0 ja oikean alakulman 255,191.
Mukana ovat myös grafiikan apuna käytettävät funktiot, kuten STICK, STRIG, jne.

BASE

MUOTO: BASE (X)

VERSIOT: MSX: KASET. + LEVYKE
SVI: ei käytössä

TARKOITUS: Funktio palauttaa taulukon X osoitteen.

LISÄYKSET: Ilmaisu X on kokonaisluku välillä 0—19, jonka vastaava merkitys on jokin seuraavista:

X		PALAUTTAA TEKSTIMOODI
0	;	palauttaa tekstimoodin (40 saraketta / 24 riviä) ensimmäisen osoitteen videomuistissa. Ilman muutoksia palauttaa ilmaisu osoitteen 0000 (taulukon loppuosoite on normaalisti 00959).
1	;	ei merkitystä.
2	;	palauttaa osoitteen, josta alkaa näppäinten merkkikooditaulukko. Muuttamalla datoja tällä alueella, voidaan näppäinten merkitykset vapaasti määritellä (ts. voidaan esim. vaihtaa skandinaaviset näppäimet). Jokaista näppäintä kohti on varattu kahdeksan (8) muistipaikkaa.
3	;	ei merkitystä.
4	;	ei merkitystä.

LEVENNETTY TEKSTIMOODI

5	;	palauttaa levennetyn tekstimoodin (32 saraketta * 24 riviä) ensimmäisen osoitteen videomuistissa. Ilman muutoksia palauttaa ilmaisu arvon 06144 (vrt. BASE(0)-funktioon).
6	;	palauttaa levennetyn tekstimoodin värikooditaulukon alkuosoitteen.
7	;	palauttaa osoitteen, josta alkaa näppäinten merkkikooditaulukko. Muuttamalla datoja tällä alueella, voidaan näppäinten merkitykset vapaasti määritellä (HUOM! EI vaikuta SCREEN 0 -moodiin).
8	;	palauttaa SPRITE-kuvahahmojen tunnustaulukon alkuosoitteen tekstimoodissa.
9	;	palauttaa SPRITE-hahmojen datataulukon alkuosoitteen.

TARKKUUSGRAFIIKKAMOODI

10	;	palauttaa grafiikkamoodin taulukon alkuosoitteen (alkaen kuvaruudun vasemmasta yläkulmasta eli pisteestä 0,0). Ilman muutoksia osoite 06144 eli sama, kuin levennetyn tekstitilanmoodi (ilmaisu 5).
11	;	palauttaa värikooditaulukon alkuosoitteen grafiikkamoodissa.
12	;	palauttaa kuva-alkioiden taulukon alkuosoitteen. Datat ovat 8-bittisiä ja järjestys on alkaen kuvaruudun vasemmasta yläkulmasta. Vrt. ilmaisuun kaksi (2).
13	;	palauttaa SPRITE-hahmojen datojen sijainnin videomuistissa. Ilman muutoksia 6912.
14	;	palauttaa SPRITE-hahmojen mallien taulukon alkuosoitteen.

LOHKOGRAFIIKKAMOODI

15	;	palauttaa puoligrafiikkatilassa käytettävien merkkien merkkikooditaulukon ensimmäisen osoitteen. Ilman muutoksia 2048.
16	;	ei merkitystä.
17	;	palauttaa kuva-alkioiden taulukon alkuosoitteen lohkografiikkamoodissa. Vrt. ilmaisuun 12.
18	;	palauttaa SPRITE-hahmojen datojen sijainnin videomuistissa.
19	;	palauttaa SPRITE-hahmojen mallien taulukon alkuosoitteen.

Mikäli muutoksia taulukoiden ilmoittamille alueille VPOKE-käskyllä tehdään, tulee muistaa, että muutamat käskyt, kuten esim. SCREEN, WIDTH, jne. palauttavat alkuperäisarvot.

ESIM:

```
PRINT BASE (0) <ENTER>
  0
Ok
LET A% = BASE (2) <ENTER>
Ok
PRINT A% <ENTER>
  2048
Ok
```

CIRCLE

MUOTO: **CIRCLE (X,Y),<säde> [,<väri>] [,<alku>] [,<loppu>] [,<suhde>]**

VERSIOT: **MSX: KASET. + LEVYKE**
SVI: KASET. + LEVYKE

TARKOITUS: Käskyllä ellipsi, ympyrä tai sen osa.

LISÄYKSET: Ilmaisulla X ja Y annetaan ellipsin (ympyrä on ellipsin äärimmäistapaus) keskipisteen koordinaatit. Ilmaisun X tulee olla välillä 0—255 ja ilmaisun Y välillä 0—191, jos keskipisteen halutaan olevan kuvaruudun näkyvällä osalla.
Ilmaisut X ja Y voidaan antaa myös negatiivisina lukuina tai suurempina, kuin edellä on esitetty, jolloin keskipiste tulostuu kuvaruudun ulkopuolelle ja ellipsin kehästä jää vain osa näkyviin.
<säde> = ympyrän säteen pituus. Ilmaisu on kokonaisluku välillä 0—32767, tosin jos ilmaisun <säde> arvo on yli 95, ei ympyrä tulostu täydellisenä, vaan osa ympyrän kehästä jää kuvaruudun ulkopuolelle. Jos säteen arvoksi annetaan nolla (0), tulostuu pelkkä piste (.).
Ympyrää/ellipsiä piirrettäessä ovat pakollisia ilmaisuja vain keskipisteen koordinaatit ja säde.
<väri> = väri, jolla ympyrän tai ellipsin kehä piirretään. Jos ilmaisua EI anneta, tulostuu kehä voimassa olevalla tekstin värillä (kts. COLOR-käsky). Ilmaisu on kokonaisluku väliltä 0—15.
<alku> = piste ympyrän ajatellulla kehällä, josta piirto aloitetaan. Tämän ilmaisun avulla voidaan ympyrän kehästä tulostaa vain osa. Ilmaisun oletusarvo on ellipsin ensimmäisen neljänneksen (kello kolme) X-akselilla. Ilmaisun tulee olla välillä 0—6.28. Arvo nolla on siis oletusarvo ja arvolla 6.28 tulostuu piste, koska ympyrän kehän pituus on 6.28 (2*pii).
<loppu> = piste ympyrän ajatellulla kehällä, johon piirto lopetetaan. Oletusarvona on sama, kuin ilmaisulla >alku> eli "kello kolme", tosin lukuna 6.28. Jos arvoksi määritellään pienempi luku, kuin 6.28, jää osa ellipsin kuvaajasta tulostumatta.
<suhde> = korjauskerroin, jolla voidaan vaikuttaa ellipsin muotoon. Jos ilmaisu on <1, piirtyy vaakatason suuntainen ellipsi ja jos ilmaisu on > 1, piirtyy pystysuunnassa oleva ellipsi. Ilmaisu voidaan antaa desimaali-, murto- tai kokonaislukuna. Käytettäessä TV-vastaanotinta tulee korjauskertoimeksi <suhde> määritellä luku 4/3, jos halutaan tulostaa tarkka ympyrä.

ESIM:

```
10 SCREEN 2
20 CIRCLE (128,96),50
30 GOTO 30
```

SVI-BASIC:ssä tulee rivi 10 antaa : '10 SCREEN 1'. Rivillä 10 annetaan käsky siirtyä tarkkuusgrafiikkaan (huom! ero SVI- ja MSX-BASIC:n välillä). Rivillä 20 piirretään ympyrä, jonka keskipisteeksi annetaan koordinaatit (128, 96) ja säteeksi 50. Rivin 30 GOTO-käsky tekee silmukan, jonka voi pysäyttää vain painamalla CTRL + STOP. Jos käytössäsi on TV-vastaanotin, tulostuu ellipsi. Asian voit korjata kirjoittamalla rivin 20 seuraavasti:

```
20 CIRCLE (128,96),50,,,,4/3
```

Jos ilmaisu <alku> tai <loppu> annetaan negatiivisena lukuna, piirtää käsky viivan ajatellusta keskipisteestä alku- tai loppupisteeseen ja arvoja <alku> ja <loppu> käsitellään, kuin ne olisivat positiivisia lukuja. Tällä tavoin voi helposti toteuttaa piirakkadigrammeja.
Seuraavaksi esimerkkejä ympyröistä ja sen osien piirtämisestä. Voit kokeilla niitä sijoittamalla käskyn edellisen ohjelman rivin 20 CIRCLE-käskyn paikalle.

KÄSKY		PIIRTÄÄ
CIRCLE (128,96),60,15	;	koko ellipsin keskelle kuvaruutua. Säde = 60 ja kehän väri valkoinen (15).
CIRCLE (128,96),60,15,3.14	;	ellipsin kolmannen ja neljännen neljänneksen, koska arvolla 3.14 (eli pii) määrätään piirto aloitettavaksi pii:n päästä normaalista aloituspisteestä.
CIRCLE (128,96),60,15,,3.14	;	ensimmäisen ja toisen neljänneksen.
CIRCLE (128,96),60,15,,6.28	;	koko ellipsin. Vastaa ensimmäistä esimerkkiä.
CIRCLE (128,96),60,15,6.28	;	pisteen.
CIRCLE (128,96),60,15,,6.28	;	täyden ellipsin.
CIRCLE (128,96),60,15,,1.57	;	I:sen neljänneksen.
CIRCLE (128,96),60,,1.57,3.14	;	II:n neljänneksen.
CIRCLE (128,96),60,,3.14,4.71	;	III:n neljänneksen.
CIRCLE (128,96),60,,4.71	;	IV:n neljänneksen.
CIRCLE (128,96),60,,1.57,4.71	;	II ja III neljännes.
CIRCLE (128,96),60,,4.71,1.57	;	IV ja I neljännes.

```
CIRCLE (128,96),60,,,,1/3     ; vaakatason suuntainen ellipsi.
CIRCLE (128,96),60,,,,5/3     ; pystyasentoinen ellipsi.
CIRCLE (128,96),60,,,,100/1   ; vaakasuora viiva.
CIRCLE (128,96),60,,,,1/100   ; pystysuora viiva.
```

Lopuksi esimerkkiohjelma CIRCLE-käskyn käytöstä piirakkadiagrammojen tulostuksessa.

```
10  SCREEN 2
20  CIRCLE (128,96),60,6,—.01,—.99
30  CIRCLE (128,96),60,10,—1—,2.38
40  CIRCLE (128,96),60,15,—2.4,—5
50  CIRCLE (128,96),60,12,—5.01,—6.27
60  GOTO 60
```

Rivi 10 tulee SVI-BASIC:ssä kirjoittaa seuraavasti:

```
10  SCREEN 1
```

CLS

MUOTO: **CLS**

VERSIOT: **MSX: KASET. + LEVYKE**
SVI: KASET. + LEVYKE

TARKOITUS: Käskyllä tyhjennetään kuvaruutu kaikesta tekstistä ja/tai grafiikasta.

LISÄYKSET: Käskyn käyttö ei poista tekstimuodossa kuvaruudun alareunassa näkyviä funktionäppäinten merkityksiä.

Käsky voidaan antaa komentotilassa:

CLS <ENTER>

tai kaikissa SCREEN-muodoissa ohjelmallisesti:

10 CLS

SCREEN- ja WIDTH-käskyt sisältävät automaattisen CLS-komennon.
Komentotilassa voidaan käsky korvata painamalla CTRL + L tai CLS-näppäintä (SPECTRAVIDEO) tai SHIFT + CLS/HM -näppäimiä (MSX).
Ohjelmallisesti voidaan CLS-käsky korvata lauseella:

ESIM:

```
10 PRINT CHR$ (12)        ; vastaa CLS-käskyä.
```

COLOR

MUOTO: **COLOR** [<**teksti**>] [,<**tausta**>] [,<**rajat**>]

VERSIOT: **MSX: KASET. + LEVYKE**
SVI: KASET. + LEVYKE

TARKOITUS: Käskyllä vaihdetaan tekstin, taustan ja/tai grafiikkamuodossa näkyvien reunojen väriä.

LISÄYKSET: Kerrallaan voidaan vaihtaa kaikkien, kahden tai yhden ilmaisun väriä. Jokainen merkki kuvaruudulla muodostuu kahdesta väristä: teksti- ja taustaväristä. Tekstiväriä on merkki itse ja taustaväri on merkin ympärille muodostuva ''laatikko''. Koska ''laatikot'' ovat toisissaan kiinni, muodostavat ne yhdessä taustan.
Ilmaisulla <rajat> tarkoitetaan grafiikkamuodoissa (SCREEN 2 ja 3 MSX-BASIC:ssä ja SCREEN 1 ja 2 SVI-BASIC:ssä) kuvaruudun ylä- ja alareunaan jääviä alueita, joille ei voida tulostaa.
Oletusarvot:

MSX-BASIC:	COLOR 15,4,4
SVI-BASIC:	COLOR 15,4,5

Käytettävissä olevat 16 väriä ovat:

VÄRIN KOODI	**VÄRI**
0	LÄPINÄKYVÄ
1	MUSTA
2	VIHREÄ

3	VAALEANVIHREÄ
4	TUMMANSININEN
5	VAALEANSININEN
6	TUMMANPUNAINEN
7	TURKOOSI
8	PUNAINEN
9	VAALEANPUNAINEN
10	TUMMANKELTAINEN
11	VAALEANKELTAINEN
12	TUMMANVIHREÄ
13	VIOLETTI
14	HARMAA
15	VALKOINEN

ESIM:

KÄSKY		**VAIHTAA VÄRIT**
COLOR 7,1	;	taustan mustaksi ja tekstin turkoosiksi.
COLOR 15	;	tekstin valkoiseksi (ei muuta taustan ja rajojen värejä).
COLOR ,8	;	taustan punaiseksi.
COLOR ,,1	;	vaihtaa grafiikkamuodossa jäävien reunojen värit mustaksi (huomioi, että vaikutus näkyy vasta siirryttäessä seuraavan kerran grafiikkamuotoon).

COLOR-käskysana löytyy funktionäppäimestä yksi (1) ja perusvärit palauttava COLOR-käsky funktionäppäimestä 6.
Värejä yhdistelemällä voidaan kehittää uusia värejä vaikkapa seuraavasti:

```
10 SCREEN 2
20 FOR A%=40 TO 80 STEP 2
30 LINE (30,A%)—(50,A%),7
40 NEXT
50 FOR A%=41 TO 79 STEP 2
60 LINE (30,A%)—(50,A%),8
70 NEXT
80 GOTO 80
```

Rivi 10 on muutettava SVI-BASIC:ssä seuraavaksi:

```
10 SCREEN 1
```

Ohjelma piirtää laatikon kahdella erivärisellä viivalla ja toimii seuraavasti. Rivillä 10 siirrytään tarkkuusgrafiikkamuotoon. Rivit 20—40 muodostavat ensimmäisen silmukan, jonka aikana piirretään viivoja joka toisen pikselin välein, värillä 7. Rivit 50—70 muodostavat toisen silmukan, joka piirtää viivoja edellisessä silmukassa piirrettyjen viivojen väliin värillä 8. Näin syntyy uusi väri. Rivin 80 GOTO-käsky on päättymätön silmukka, jonka voi pysäyttää painamalla CTRL + STOP.
Värit ilmaistaan siis aina numeroilla, ei kirjaimilla. Monet grafiikkakäskyt sisältävät mahdollisuuden valita tulostuksessa käytettävää väriä, jolloin käytetään edellä esitettyä värikoodien listaa.

DRAW

MUOTO: **DRAW <string>**

VERSIOT: **MSX: KASET. + LEVYKE**
SVI: KASET. + LEVYKE

TARKOITUS: Käskyllä piirretään kuvio grafiikan makrokieleen perustuen.

LISÄYKSET: Ilmaisu <string> voi olla merkkijonomuuttuja tai merkkijono, joka koostuu makrokäskyistä. Makrokäskyt muodostuvat kirjaimesta ja siihen liittyvästä luvusta. Kirjaimella ilmaistaan suunta tai kohta, josta piirto aloitetaan ja luku ilmaisee piirrettävien pisteiden määrän.

Makrokäskyt ovat seuraavat:

KÄSKY		**TOIMINTA**
U <nro>	;	piirtää kursorin sen hetkisestä paikasta ilmaisun <nro> verran ylöspäin.
D <nro>	;	piirtää alaspäin.

R <nro> ; piirtää oikealle.
L <nro> ; piirtää vasemmalle.
E <nro> ; piirtää oikealle ylös.
F <nro> ; piirtää oikealle alas.
G <nro> ; piirtää vasemmalle alas.
H <nro> ; piirtää vasemmalle ylös.

Ilmaisulla <nro> annetaan siis määrä, joka liikutaan haluttuun suuntaan. Voidaan myös määritellä kerroin, jolla haluttu kuvio voidaan moninkertaistaa.

S <nro> ; asetetaan kerroin (kokonaisluku väliltä 0-255), jolla piirretty kuvio voidaan muuntaan halutun kokoiseksi.

Todellinen kerroin saadaan jakamalla <nro> neljällä (4). Jos siis <nro> on esim. yksi (1), on korjauskerroin 0.25. Ilmaisun <nro> oletusarvo on neljä (4), joka vastaa käskyä 'S4'.

Tulostus (piirto) jatkuu aina pisteestä, johon on viimeksi jääty, ellei erikseen käytetä jotain seuraavista käskyistä:

M <X,Y> : siirtää paikkaa, josta piirto alkaa, joko absoluuttisesti tai suhteellisesti.

Jos ilmaisua X edeltää plus- (+) tai miinus- (—) merkki, siirretään paikkaa suhteellisesti eli sen hetkisiin koordinaattien X ja Y arvoihin lisätään käskyn M vastaavat luvut. Jos ilmaisua X ei edellä etumerkkiä, siirretään piirto alkamaan käskyllä M annettavista koordinaattipisteistä.
Käytettäessä M-makrokäskyä piirtyy viiva paikkaa muutettaessa kohdasta, johon piirto viimeksi jäi, käskyllä ilmotettuihin koordinaattipisteisiin. Jos tältä halutaan välttyä, lisätään M-käskyn eteen B-kirjain.

BM <X,Y> : siirtää paikkaan, josta piirto alkaa, joko absoluuttisesti tai suhteellisesti.

Käsky toimii samoin, kuin pelkkä M-käsky, mutta tällä kertaa ei siirrettäessä piirretä viivaa. Jos piirron jälkeen halutaan palata takaisin paikkaan, josta piirto alkaa, tulee käyttää komentoa:

NM <X,Y> ; siirtää paikkaan, josta piirto alkaa (pisteeseen X,Y), suorittaa piirron ja palauttaa lopuksi takaisin koordinaattipisteeseen, josta piirto alkoi (ensimmäisen DRAW-käskyllä määriteltyyn kohtaan).

Voidaan antaa suhteellisena tai absoluuttisena.
Kesken piirron voidaan muuttaa kulmaa käskyllä:

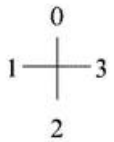
A <nro> ; kääntää tulostusta portaittain 90:n asteen välein.

Ilmaisu <nro> voi saada seuraavat arvot:

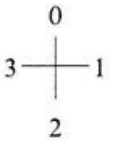
0 ----> nolla (0) astetta.
1 ----> 90 astetta.
2 ----> 180 astetta.
3 ----> 270 astetta.

Suunnat eroavat MSX- ja SVI-BASIC:ssä seuraavasti:

MSX



SVI



Väriä voidaan säätää C-käskyllä.

C <nro> ; muuttaa väriksi ilmaisun <nro> antaman värikoodin.

Ilmaisu <nro> voi olla välillä 0—15 (kts. COLOR-käsky).
Piirrettävä kuvio voidaan antaa myös merkkijonon sisältä toisella merkkijonolla. Tällä tavoin voidaan yhteen DRAW-funktioon sisällyttää yli 255 merkkiä.

X <string> ; piirtää merkkijonon <string> määrittelemän kuvion.

Merkkijonoa <string> seuraa puolipiste (;), jolla ilmaistaan merkkijonon päättyneen.
Kaikissa makrokäskyissä olevat ilmaisut X, Y ja <nro> voidaan korvata myös numeerisella muuttujalla, jolloin muuttujaa edeltää yhtäsuuruusmerkki (=) ja perään lisätään puolipiste (;).

```
A%=20 : B%=30 : DRAW "F=A%; D=B%;"
```

Tämä esimerkki piirtää ensin alas oikealle 20 yksikköä eli pikseliä ja sitten alas 30 yksikköä.
Välilyönnit (tyhjä) eivät vaikuta merkkijonojen (muuttujien) sisällä, joten seuraavat kaksi esimerkkiä ovat identtisiä:

```
DRAW "R20D20"          DRAW "R20 D20"
```

ESIM:

```
10  SCREEN 2
20  A$ ="R20 F20 D20 G20 L20 H20 U20 E20"
30  DRAW "BM120,60 S4 XA$;"
40  DRAW "BM+5, 15 S2 XA$;"
50  GOTO 50
```

Rivi 10 on SVI-BASIC:ssä kirjoitettava seuraavasti:

```
10  SCREEN 1
```

Rivillä 10 siirrytään tarkkuusgrafiikkaan. Rivillä 20 määritellään merkkijonomuuttujalle A kuvio (8-kulmio), joka riveillä 30 ja 40 tulostetaan DRAW-käskyllä. Rivillä 30 kohdistetaan piirto alkavaksi koordinaattipisteistä (120, 60), asetetaan korjauskertoimeksi 4 (S4-käsky) ja annetaan tulostettava kuvio merkkijonona. Rivillä 50 suoritetaan siirto pisteeseen, josta piirto alkaa, suhteellisesti lisäämällä viimeksi piirrettyyn pisteeseen X-koordinaattiin luku 5 ja Y-koordinaattiin luku 15. Korjauskertoimeksi asetetaan 2. Tulostettava kuvio on sama, kuin rivillä 30 määritellään, tosin puolet pienempi. Rivin 50 GOTO-käsky sulkee ohjelman suorituksen ikuiseen silmukkaan, jonka voi pysäyttää painamalla CTRL+STOP -näppäimiä.
Seuraava esimerkki piirtää satunnaisesti kolmiulotteisia neliöitä. Jos käytössäsi on SVI-BASIC, kirjoita rivi 10 seuraavasti: 10 SCREEN 1.

```
10  SCREEN 2
20  A$="D80 R80 E30 L80 U80 R80 G30 L80 E30 R80 D80 G30 U80 L80 D80 E30"
30  A%=INT (255*RND (1)+1)
40  B%=INT (191*RND(1)+1)
50  C%=INT (2*RND(1)+1)
60  D%=INT (3*RND(1)+1)
70  DRAW "S=C%; BM =A%;, =B%; A=D%; XA$;"
80  GOTO 30
```

Rivillä 10 siirrytään tarkkuusgrafiikkaan. Rivillä 20 määritellään piirrettävän kuvion muoto (neliö). Rivillä 30 arvotaan X-koordinaatti, josta neliön piirto alkaa ja rivillä 40 arvotaan Y-koordinaatin aloituspiste. Rivillä 50 arvotaan korjauskerroin, jolla määrätään neliön koko. Rivillä 60 arvotaan kulma, jossa neliö piirretään (0—360 astetta). Rivillä 70 tapahtuu varsinainen piirto, jossa annetaan ensin koko, aloituspisteet, kulma ja lopuksi itse kuvio (XA$;). Riviltä 80 hypätään takaisin riville 30 ja seuraavan neliön tietojen arpominen alkaa. Ohjelman voi pysäyttää hetkeksi STOP-näppäimestä ja kokonaan CTRL+STOP.

GET

MUOTO: **GET (X1,Y1)—(X2,Y2),<nimi>**

VERSIOT: **SVI: KASET. + LEVYKE**
MSX: ei käytössä

TARKOITUS: Käskyllä luetaan kuvaruudulta ilmaisuilla (X1, Y1) ja (X2, Y2) määritelty alue matriisiin <nimi> sisällöksi.

LISÄYKSET: Ilmaisu (X1,Y2) määrittelee luettavan alueen vasemman yläkulman koordinaatit ja ilmaisu (X2,Y2) alueen oikean alakulman koordinaatit.
GET-käsky lukee määritellyn alueen kaikki pisteet, mukaanlukien reuna-alueet, väreineen matriisin <nimi> sisällöksi. Ennen GET-käskyn käyttöä on matriisille varattava tilaa DIM-käskyllä. Varattavan tilan suuruus lasketaan seuraavasti:

(X2—X1)/2 = A ja (Y2—Y1)/2 = B

Varattava matriisi on tällöin DIM <nimi>(A,B).
Ilmaisu <nimi> on mikä tahansa SVI-BASIC:n hyväksymä muuttujanimi.
GET-käskyllä luettu alue voidaan myöhemmin piirtää PUT-käskyllä minne tahansa kuvaruudulla. Tosin piirrettävä kuva EI saa ylittää Y-koordinaattia 191 eikä alittaa Y-koordinaattia nolla (0).
GET-käsky ei poista kuvaruudulta kuvaa tai sen osaa, joka sillä luetaan.

Matriisi <nimi> on hyödyllistä määritellä kokonaislukumatriisiksi, koska tiedot luettavista pisteistä taltioidaan kokonaislukuina. Kokonaislukumatriisin varaama RAM-muistin määrä on pienin mahdollinen ja näin jää varsinaiselle ohjelmalle mahdollisimman paljon tilaa.

ESIM:

```
10  SCREEN 1
20  DEFINT A
30  DIM A(28,27)
40  REM piirretään kuva
50  CIRCLE (128,96),20,15
60  CIRCLE (128,98),15,15,3.14
70  PSET (121,90)
80  PSET (122,90)
90  PSET (134,90)
100 PSET (135,90)
110 PSET (128,95)
120 PSET (128,96)
130 REM piirretään kehykset
140 LINE (100,70)—(156,124),15,B
150 REM luetaan piirretty alue kehyksineen muistiin
160 GET (100,70)—(156,124),A
170 REM piirretään samanlaisia kuvia näytölle
180 PUT (10,10),A,PSET
190 PUT (10,70),A,PSET
200 PUT (10,130),A,PSET
210 PUT (100,10),A,PSET
220 PUT (100,130),A,PSET
230 PUT (190,10),A,PSET
240 PUT (190,70),A,PSET
250 PUT (190,130),A,PSET
260 GOTO 260
```

Rivillä 10 siirrytään tarkkuusgrafiikkaan. Rivillä 20 määritellään kaikki A-kirjaimella alkavat muuttujat kokonaislukumuuttujiksi. Rivillä 30 varataan matriisille A tarvittava tila GET-käskyä varten. Riveillä 40, 130, 150 ja 170 on listausta selventäviä huomautuksia. Riveillä 50—120 piirretään kuva käyttäen CIRCLE- ja PSET-käskyjä. Rivillä 140 piirretään kuvan ympärille kehykset, jotta myöhemmin voidaan GET-käskyn lukevan myös alueen reunat matriisin sisällöksi. Rivillä 160 luetaan kuva matriisin muistiin ja riveillä 180—250 piirretään sama kuva useaan paikkaan. Rivin 260 GOTO-käsky sulkee ohjelman silmukkaan, jonka voi pysäyttää painamalla CTRL+STOP.

LINE

MUOTO: **LINE [<koord1>] [— <koord2>] [,<väri>] [,B/BF]**

VERSIOT: **MSX: KASET. + LEVYKE**
SVI: KASET. + LEVYKE

TARKOITUS: Käskyllä piirretään kahden pisteen välille suora.

LISÄYKSET: <koord1> = ilmaisulla määritellään piste, josta viivan piirto aloitetaan. Ilmaisu on muotoa (X,Y), jossa X on X-akselin ja Y on Y-akselin koordinaattipiste. Ilmaisu voidaan jättää antamatta, jolloin piirto aloitetaan pisteestä, johon viimeksi on piirretty jollain grafiikkakäskyllä (huomioi poikkeustapaukset DRAW-käskyn suhteen). Oletusarvona pisteelle on (0,0) eli kuvaruudun vasen yläkulma.
<koord2> = ilmaisulla määritellään piste, johon piirto lopetetaan. Ilmaisun muoto on sama, kuin pisteen <koord1>.
Sekä piste <koord1> että <koord2> voidaan antaa absoluuttisena (määrätty piste kuvaruudulla) tai suhteellisena, jolloin muoto on seuraava:

STEP (A,B)

Nyt pisteen koordinaateiksi tulee arvot, jotka saadaan lisäämällä X-koordinaattiin (johon viimeksi on piirretty) luku A ja vastaavasti Y-koordinaattiin luku B. Lisäys A ja B voi olla välillä 0—255 (jos arvo on 0, ei STEP:llä ole merkitystä).
Seuraavat kaksi esimerkkiä vastaavat toisiaan:

LINE (10,10)—STEP (20,30)
LINE (10,10)—(30,40)

Jos pisteen <koord1> tai <koord2> X-piste ei ole välillä 0—255 tai Y-piste välillä 0—191, suoritetaan pyöristys lähinnä seuraavaan pisteeseen grafiikka-alueen reunassa. Arvojen X ja Y tulee olla joka tapauksessa välillä —32768 — +32767.

<väri> = värikoodi, jota vastaavalla värillä viiva piirretään. Oletusarvona on viimeksi käytössä ollut tekstin väri (kts. COLOR-käsky).
LINE-käskyllä voidaan piirtää myös suorakulmio, jolloin lauseen loppuun lisätään ilmaisu ',B'. Tällöin piste <koord1> ilmaisee suorakulmion vasemman yläkulman koordinaattipisteet ja vastaavasti piste <koord2> suorakulmion oikean alakulman koordinaattipisteet.
Jos piirretty suorakulmio halutaan samalla täyttää jollain värillä, lisätään lauseen perään ilmaisu 'BF'. Väri määritellään ilmaisulla <väri>.
Ilmaisujen X ja Y arvot voidaan antaa myös negatiivisina, jolloin piirtäminen (tai siirtyminen STEP:n avulla) tapahtuu vasemmalle (negatiivinen X-koordinaatin arvo) ja ylöspäin (negatiivinen Y-koordinaatin arvo).

ESIM:

```
10  SCREEN 2
20  LINE (10,10)—30,30)
30  LINE —(100,130),7
40  LINE STEP (40,-80)—STEP (50,50),15,BF
50  LINE (30,150)—(210,180),,B
60  GOTO 60
```

Rivi 10 on kirjoitettava SVI-BASIC:ssä seuraavasti:

```
10  SCREEN 1
```

Rivillä 10 siirrytään tarkkuusgrafiikkaan. Rivillä 20 annetaan käsky piirtää viiva pisteiden (10,10) ja (30,30) välille. Rivi 30 piirtää viivan, alkaen pisteestä (30,30), pisteeseen (100,130), värillä 7 (turkoosi). Rivi 40 piirtää väritetyn suorakulmion (BF), alkaen pisteestä (140,50) (pisteen koordinaatit saadaan seuraavasti: 100+40 = 140 ja 130—80 = 50), asettaen oikean alakulman pisteeseen (190,100), jonka koordinaatit saadaan seuraavasti:

140+50 = 190 ja 50+50 = 100

Rivi 50 piirtää suorakulmion kuvaruudun alareunaan, käyttäen oletusarvoväriä. Rivin 60 GOTO-käsky aikaansaa ikuisen silmukan, jonka voi pysäyttää painamalla CTRL+STOP.

LOCATE

MUOTO:

MSX: LOCATE [X] [,Y]
SVI: LOCATE [X] [,Y] [,Z]

VERSIOT:

MSX: KASET. + LEVYKE
SVI: KASET. + LEVYKE

TARKOITUS:

Käskyllä kohdistetaan tulostus haluttuun kohtaan kuvaruudulla

LISÄYKSET:

SVI-BASIC sallii käskyn käytön kaikissa kuvaruutumoodeissa, kun taas MSX-BASIC:ssä LOCATE-käskyä voidaan käyttää vain tekstimoodissa (SCREEN 0 tai SCREEN 1).

X = sarake, johon kursori siirretään.
Y = rivi, jolle kursori siirretään.

Sekä ilmaisun X että Y oletusarvo on nolla (0).
LOCATE-käsky siirtää kursorin aina yhtä saraketta oikeammalle, kuin X:n arvo edellyttää, sillä sarakkeet on numeroitu kuvaruudulla luvusta nolla (0) alkaen ja normaalisti käyttäjä laskee sarakkeita luvusta yksi (1) alkaen. Siksi esimerkiksi käsky 'LOCATE 0,10' siirtää kursorin sarakkeeseen yksi, riville 10. Eli jokaisen rivin ensimmäinen sarake on numeroltaan nolla (0), vaikka me näemme sen sarakkeena yksi (1).
Ilmaisulla Z saadaan SVI-BASIC:ssä kursori pois näkyvistä. Tätä ominaisuutta on hyödyllistä käyttää ohjelmissa, joissa tulostetaan paljon tekstiä kuvaruutuun, sillä estämällä kursorin näkyminen saadaan tulostukseen lisää nopeutta ja tekstin "välkkyminen" eliminoituu. Esimerkiksi käsky:

```
LOCATE,,0      ; poistaa kursorin kuvaruudulta.
LOCATE,,1      ; asettaa kursorin näkyviin.
```

Ilmaisut X ja Y voivat saada seuraavia arvoja:

X

MSX-BASIC	SVI-BASIC		LISÄYKSET
0—39	0—39	;	SCREEN 0 (WIDTH 40)
0—31		;	SCREEN 1 (WIDTH 32)
	0—255	;	SCREEN 1 (tarkkuusgrafiikkamoodi)
	0—255	;	SCREEN 2 (lohkografiikkamoodi)

Y

MSX-BASIC	SVI-BASIC		LISÄYKSET
0—22	0—22	;	SCREEN 0 (funktionäppäimet näkyy)
0—23	0—23	;	SCREEN 0 (funktionäppäimet eivät näy)
0—22		;	SCREEN 1 (funktionäppäimet näkyy)
0—23		;	SCREEN 1 (funktionäppäimet eivät näy)
0—191		;	SCREEN 1 (tarkkuusgrafiikkamoodi)
0—191		;	SCREEN 2 (puoligrafiikkamoodi)

Jos käytössä on 80 merkkiä/rivi -laajennusyksikkö, saadaan jokaiselle riville käyttöön 80 saraketta. Tällöin ovat sarakkeet numeroitu 0—79 ja käytössä on vain tekstimoodi.
Jos X:lle annetaan suurempi arvo, kuin edellä olevassa taulukkossa on määritelty, jää kohdistin (kursori) rivin viimeiseen sarakkeeseen ja jos Y:lle annetaan suurempi arvo, kuin taulukossa on esitetty, jää kohdistin viimeiselle riville. Negatiivisia arvoja ei hyväksytä.

ESIM:

```
10 CLS
20 LOCATE 12,0
30 PRINT "TEKSTINKÄSITTELY"
40 LOCATE 0,4
50 PRINT "COPYRIGHT (C) MICROSOFT"
60 LOCATE 60,10
70 PRINT "12/10/1995"
```

Rivillä 10 tyhjennetään kuvaruutu. Rivillä 20 asetetaan kursori riville 0, sarakkeeseen 13 (12+1), josta alkaen rivin 30 teksti "TEKST..." tulostuu. Rivillä 40 asetetaan kohdistin riville 4, sarakkeeseen 1 (0+1), josta alkaen rivin 50 teksti tulostuu. Rivillä 60 siirretään kursori riville 10, sarakkeeseen 39 (38+1, ellei WIDTH-komentoa ole käytetty ennen ohjelman ajoa ja ellei 80 merkkiä/rivi -laajennusyksikkö ole käytössä), sillä kohdistin pysähtyy rivin loppuun, vaikka sarakkeeksi on merkitty nro 60. Rivillä 70 tulostetaan teksti, jonka ensimmäinen merkki (1) tulostuu riville 10 ja loput riviltä 11, sarakkeesta 1 alkaen.

PAD

MUOTO: **PAD (X)**

VERSIOT: **MSX: KASET. + LEVYKE**
SVI: KASET. + LEVYKE

TARKOITUS: Funktio palauttaa piirtoalustan (grafiikkapiirtolevyn) rekisterin X tilan/arvon.

LISÄYKSET: Funktio eroaa MSX- ja SVI-BASIC:ssä siten, että MSX-BASIC sallii kahden piirtoalustan käytön samanaikaisesti, kun taas SVI-BASIC yhden.
Ilmaisulla X tarkoitetaan piirtoalustan rekisteriä, joka on jokin seuraavista:

X	VERSIO		TARKOITTAA
0	MSX ja SVI	;	peliohjainporttiin yksi kytketyn piirtolevyn tilarekisteriä. Palauttaa arvon nolla (0) eli EPÄTOSI, jos piirtolevyä EI kosketeta ja arvon yksi (1) eli TOSI, jos piirtolevyä on painettu.
1	MSX ja SVI	;	porttiin yksi kytketyn piirtolevyn X-akselin tietorekisteri. Palauttaa painetun pisteen X-koordinaatin väliltä 0—255.
2	MSX ja SVI	;	porttiin yksi kytketyn piirtolevyn Y-akselin tietorekisteri. Palauttaa painetun pisteen Y-koordinaatin väliltä 0—191.
3	MSX ja SVI	;	porttiin yksi kytketyn piirtolevyn tilarekisteriä piirron aikana. Palauttaa luvun —1, jos piirtoalustaa painetaan ja muussa tapauksessa nollan (0).
4	MSX	;	sama, kuin ilmaisu nolla, mutta peliohjainporttiin kaksi (2) kytkettyä piirtoalustaa.
5	MSX	;	sama kuin ilmaisu yksi, mutta piirtoalusta on kytkettynä porttiin kaksi (2).
6	MSX	;	sama, kuin ilmaisu kaksi, mutta piirtoalusta on kytkettynä porttiin kaksi (2).
7	MSX	;	sama, kuin ilmaisu kolme, mutta piirtoalusta on kytkettynä porttiin kaksi (2).

Huomaa, että ennen painetun pisteen koordinaattien X ja Y saamista, on annettava PAD (0) tai PAD (4) käsky.

ESIM:

```
10  SCREEN 1
20  IF PAD (0) THEN 30 ELSE 20
30  X%=PAD (1) : Y%=PAD (2)
40  PSET (X%,Y%)
50  GOTO 20
```

Esimerkki yksinkertaisesta piirto-ohjelmasta, joka on kirjoitettu SPECTRAVIDEO:lle. Käytettäessä MSX-BASIC:ä tulee muuttaa rivi 10 seuraavaksi:

```
10  SCREEN 2
```

Rivillä 10 siirrytään tarkkuusgrafiikkamuotoon. Rivillä 20 luetaan peliohjainporttiin yksi kiinnitetyn piirtoalustan tilarekisteri. Jos funktio palauttaa arvon -1 eli TOSI, siirrytään riville 30, muuten palataan riville 20 ja luetaan tilarekisteri uudestaan. Rivillä 30 luetaan muuttujille X% ja Y% koordinaattirekistereiden arvot. Rivillä 40 piirretään piste saatujen koordinaattien perusteella.
HUOM! kiinnitä/irrota piirtoalusta aina POWER OFF -tilassa.

PAINT

MUOTO:

MSX: **PAINT <koord> [,<väri1> [,<väri2>]**
SVI: **PAINT <koord> [,<väri1>]**

VERSIOT:

MSX: **KASET. + LEVYKE**
SVI: **KASET. + LEVYKE**

TARKOITUS:

Käskyllä täytetään alue värillä <väri1> alkaen pisteestä <koord>.

LISÄYKSET:

<koord> = piste alueen sisällä, josta täyttö halutulla värillä alkaa. Alueen tulee olla rajattu tai väritys suoritetaan koko kuvaruutuun. Ilmaisu voidaan antaa joko absoluuttisena, jolloin muoto on:

PAINT (X,Y)

jossa X ja Y ovat koordinaattipisteet, josta väritys alkaa tai suhteellisena, jolloin muoto on:

PAINT STEP (X,Y)

Nyt pisteen koordinaatit, josta väritys alkaa, saadaan lisäämällä viimeksi piirretyn pisteen X ja Y koordinaatteihin vastaavasti STEP-lauseen arvot X ja Y. Käytettäessä PAINT STEP -lausetta eivät lopullisen pisteen koordinaatit X tai Y saa mennä kuvaruudun ulkopuolelle. STEP-lauseen koordinaatit X ja Y voidaan antaa myös negatiivisina lukuina, jolloin STEP-lauseen koordinaatit vähennetään viimeksi piirretyn pisteen koordinaateista.
<väri1> = värin vastaava koodinumero (kts. COLOR-käsky), jolla väritys suoritetaan. Voidaan jättää antamatta, jolloin väritys suoritetaan oletusarvovärillä (tekstin väri, jota viimeksi on käytetty).
<väri2> = värin numero <väri2> rajaama alue, johon väritys lopetetaan. Tällä ilmaisulla voidaan antaa ehto: jos vastaan tulee raja, joka on väriltään koodia <väri2> vastaava väri, pysäytetään väritys näiden rajojen sisään, muussa tapauksessa jatketaan väritystä, kunnes vastaan tulee väri <väri1>. Tämä ilmaisu toimii vain MSX-BASIC:n SCREEN-moodissa kolme (SCREEN 3).

ESIM:

```
10  SCREEN 2
20  CIRCLE (180,90),30,15
30  CIRCLE (60,90),30,15
40  PAINT STEP (100,0),15
50  GOTO 50
```

Rivi 10 on SVI-BASIC:ssä kirjoitettava seuraavasti:

```
10  SCREEN 1
```

Rivillä 10 siirrytään tarkkuusgrafiikkamuotoon. Rivillä 20 piirretään ympyrä, jonka keskipisteeksi tulee (180,90), säteeksi 30 yksikköä ja väriksi valkoinen (15). Rivillä 30 piirretään toinen ympyrä edellisen vasemmalle puolelle. Rivin 40 PAINT STEP -lause suorittaa ensin laskutoimitukset: 100+60 = 160 ja 0+90 = 90. Saadut arvot ovat pisteen koordinaatit, josta väritys alkaa. Saatu piste (160,90) osuu oikealla olevan ympyrän sisään, joka väritetään. Jos ympyrän kehällä olisi yksikin aukko (ei pistettä), jatkaisi PAINT-käsky väritystä koko kuvaruutuun. Kokeile tätä lisäämällä edelliseen ohjelmaan rivi:

```
25  PRESET (210,90)
```

Aja ohjelma uudelleen RUN-käskyllä.

Seuraava esimerkki toimii tarkoituksen mukaisesti vain MSX-BASIC:ssä.

```
10  SCREEN 3
20  LINE (80,40)—(170,120),15,B
30  LINE (100,60)—(150,100),12,B
40  PAINT (125,80),15,12
50  GOTO 50
```

Rivillä 10 siirrytään lohkografiikkamoodiin. Rivillä 20 piirretään laatikko valkoisella värillä ja rivillä 30 toinen laatikko edellisen sisään tummanvihreällä. Rivin 40 PAINT-käsky värittää, alkaen keskeltä sisimmäistä suorakulmiota, valkoisella värillä vain sisemmän suorakulmion, koska PAINT-käskyllä on määritelty (arvo 12 lauseen lopussa) väritys pysäytettävän tummanvihreään väriin. Ilman arvoa 12 käskyn lopussa värittyisi suuremman suorakulmion sisäpuoli kokonaan. Rivin 50 GOTO-käsky sulkee ohjelman silmukkaan, jonka voi pysäyttää painamalla CTRL+STOP.
Kokeile ilmaisun <väri2> vaikutusta vaihtamalla rivi 40 seuraavaksi:

```
40 PAINT (125,80),12,15
```

Tällä kertaa värittyy uloimmainen suorakulmio.

PDL

MUOTO: **PDL (X)**

VERSIOT: **MSX: KASET. + LEVYKE**
SVI: KASET. + LEVYKE

TARKOITUS: Funktio palauttaa PADDLE-ohjaimen asennon.

LISÄYKSET: Funktio eroaa MSX- ja SVI-BASIC:n välillä siten, että ilmaisu X voi MSX-BASIC:ssä saada arvot 1—12 (parittomat arvot vastaa peliohjainporttiin yksi (1) kytkettyä PADDLE-ohjainta ja parilliset arvot porttiin kaksi (2) kytkettyä ohjainta) ja SVI-BASIC:ssä arvot 1—4 (parittomat arvot porttiin yksi kytkettyä PADDLE-ohjainta ja parilliset arvot porttiin kaksi kytkettyä). Lisäksi MSX-BASIC palauttaa arvon 255, jos ohjainta ei ole kytketty ja SVI-BASIC arvon nolla (0).
PADDLE-ohjaimella tarkoitetaan ohjainta, jossa ohjainsauvan tilalla on pyöritettävä säätöpotentiometri.
Ohjain palauttaa arvon 0-255, jossa arvo nolla (0) vastaa ohjaimen vasenta ääriasentoa ja arvo 255 ohjaimen oikeata ääriasentoa.

ESIM:

```
10  A%=PDL (1)
20  IF A% > 170 THEN PRINT "OLET KÄÄNTYMÄSSÄ OIKEALLE" :
    GOTO 10
30  IF A% < 85 THEN PRINT "OLET KÄÄNTYMÄSSÄ VASEMMALLE" :
    GOTO 10
40  PRINT "KULJET SUORAAN"
50  GOTO 10
```

Ohjelma on yksinkertainen esimerkki PADDLE-sovellutuksesta, jossa ohjaimelle on määritelty kolme mahdollista asentoa. Asennon mukaan tulostetaan, mihin suuntaan ollaan kääntymässä. Rivillä 10 luetaan PADDLE-ohjaimen asento muuttujan A% arvoksi. Rivillä 20 tutkitaan, onko ohjaimen asento yli kolmanneksen (256/3 = 85) oikealla ja jos on, tulostetaan vastaava teksti. Jos ohjaimen asento ei ollut äskeisellä alueella, siirrytään riville 30, jossa tutkitaan, onko ohjaimen asento vasemmalle painottunut ja tulostetaan vastaavasti teksti. Jos kumpikaan ehto ei toteudu, siirrytään riville 40, jossa tulostuu oma teksti. Rivien 20 ja 30 lopussa olevalla GOTO-käskyllä estetään virheelliset tulostukset ja rivin 50 GOTO-käsky palauttaa ohjelman suorituksen alkuun. Näin jatkuu, kunnes painetaan CTRL+STOP.

POINT

MUOTO: **POINT (X,Y)**

VERSIOT: **MSX: KASET. + LEVYKE**
SVI: KASET. + LEVYKE

TARKOITUS: Käsky palauttaa pisteen (X,Y) värin.

LISÄYKSET: Ilmaisun X tulee olla välillä 0—255 ja ilmaisun Y välillä 0—191, muuten palautetaan —1. Käsky toimii vain grafiikka- ja puoligrafiikkamoodissa.

ESIM:

```
10  SCREEN 2
20  LINE (30,30)—(70,70),12,BF
30  A% = POINT (60,60)
40  SCREEN 0
50  PRINT A%
```

SVI-BASIC:ssä tulee rivi 10 kirjoittaa seuraavasti:

```
10  SCREEN 1
```

Rivillä 10 siirrytään tarkkuusgrafiikkamuotoon. Rivin 20 LINE-käsky piirtää suorakulmion ja täyttää sen tummanvihreällä värillä. Tätä ei juuri ehdi nähdä, vaikka rivillä 30 ensin luetaan muuttujan A% sisällöksi pisteen (60,60) värikoodi, sillä rivillä 40 siirrytään takaisin tekstimoodiin tulostamaan muuttujan A% arvoa. Ohjelma tulostaa lopuksi luvun 12 (pisteen 60,60 värikoodi) ja palaa komentotilaan.

PRESET

MUOTO: **PRESET <koord> [,<väri>]**

VERSIOT: **MSX: KASET. + LEVYKE**
SVI: KASET. + LEVYKE

TARKOITUS: Käskyllä piirretään piste taustavärillä (ellei ilmaisulla <väri> toisin määrätä) paikkaan <koord>.

LISÄYKSET: <koord> = ilmaisu voidaan antaa absoluuttisena, jolloin muoto on:

PRESET (X,Y)

jossa X ja Y ovat pisteen koordinaatit. Piste voidaan määritellä myös suhteellisena, jolloin muoto on seuraava:

PRESET STEP (X,Y)

Nyt lopullisen pisteen koordinaatit saadaan lisäämällä viimeksi piirretyn pisteen koordinaatteihin X:n ja Y:n arvo lauseesta STEP. Jos X tai Y on negatiivinen, suoritetaan vähennyslasku.
<väri> = ilmaisulla voidaan määritellä väri, jolla piste piirretään tai jättämällä ilmaisu määrittelemättä, piirretään piste taustavärillä. Voi saada arvot 0—15 (kts. COLOR-käsky). Jos PRESET-käskyssä käytetään ilmaisua <väri>, vastaa se toiminnaltaan PSET-käskyä. Ilman ilmaisua <väri> on käsky PSET-käskyn vastakkaisfunktio.

ESIM:

```
10  SCREEN 2
20  FOR A% = 1 TO 190
30  PSET (A% + 20,A%),15
40  FOR A = 1 TO 10 : NEXT
50  PRESET (A% + 20,A%)
60  NEXT
```

SVI-BASIC:ssä on rivi 10 kirjoitettava seuraavasti:

```
10  SCREEN 1
```

Rivillä 10 siirrytään tarkkuusgrafiikkamuotoon. Rivit 20—60 muodostavat FOR-NEXT -silmukan, jonka aikana piirretään pisteitä alkaen kuvaruudun vasemmasta yläkulmasta jatkuen vinosti oikealle alas. Muuttuja A% on piirrettävän pisteen Y-koordinaatti, joka kasvaa joka kierroksella yhdellä. Rivin 40 FOR-NEXT-silmukka kuluttaa vain aikaa, jolla saadaan pisteen liikkuminen kuvaruudulla hitaammaksi. Rivin 50 PRESET-käsky sammuttaa pisteen, joka rivillä 30 sytytetään. Näin syntyy vaikutelma pisteen yhtenäisestä liikkeestä.

PSET

MUOTO: **PSET <koord> [,<väri>]**

VERSIOT: **MSX: KASET. + LEVYKE**
SVI: KASET. + LEVYKE

TARKOITUS: Käskyllä piirretään piste ilmaisun <koord> määräämään paikkaan.

LISÄYKSET: <koord> = ilmaisu voidaan antaa absoluuttisena, jolloin muoto on:

PSET (X,Y)

jossa X ja Y ovat pisteen koordinaatit. Piste voidaan määritellä myös suhteellisena, jolloin muoto on seuraava:

PSET STEP (X,Y)

Nyt lopullisen pisteen koordinaatit saadaan lisäämällä viimeksi piirretyn pisteen koordinaatteihin X:n ja Y:n arvo lauseesta STEP. Jos X tai Y on negatiivinen, suoritetaan vähennyslasku.
<väri> = ilmaisulla voidaan määritellä väri, jolla piste piirretään tai jättämällä ilmaisu määrittelemättä, piirretään piste oletusarvovärillä (viimeksi käytössä ollut tekstiväri). Ilmaisu voi saada arvot 0—15 (kts. COLOR-käsky).

ESIM:

```
10  SCREEN 2
20  A%=INT (RND(1)*255)
30  B%=INT (RND(1)*191)
40  C%=INT (RND(1)*15)
50  PSET (A,B),C
60  GOTO 20
```

Rivi 10 on SVI-BASIC:ssä kirjoitettava seuraavasti:

```
10  SCREEN 1
```

Rivillä 10 siirrytään tarkkuusgrafiikkamuotoon. Riveillä 20—40 arvotaan pisteen X- ja Y-koordinaatit sekä väri. Rivillä 50 piirretään piste käyttäen edellä määriteltyjä arvoja. Riviltä 60 palataan takaisin riville 20 arpomaan uuden pisteen koordinaatteja. Ohjelma jatkuu, kunnes painetaan CTRL+STOP.

PUT

MUOTO: **PUT (X,Y),<nimi> [,<oper>]**

VERSIOT: **SVI: KASET. + LEVYKE**
MSX: ei käytössä

TARKOITUS: Käskyllä siirretään GET-käskyllä luettu kuvaruudun osa paikkaan (X,Y).

LISÄYKSET: (X,Y) = koordinaattipisteet, joista piirto alkaa. Piste (X,Y) on piirrettävän suorakulmion vasemman yläkulman koordinaatti.
<nimi> = kuva (suorakulmio), joka on GET-käskyllä luettu kuvaruudulta matriisiksi <nimi>.
<oper> = looginen operaatio, joka suoritetaan kuvaruudulla olevan datan (kuvan) ja PUT-käskyllä annettavan matriisin <nimi> kesken. Ilmaisun oletusarvo on XOR-operaatio. Operaatiot toimintoineen ovat seuraavat:

OPERAATIO		TOIMINTA
AND	;	PUT-lauseen matriisiin datalle ja kuvaruudulla olevan kuvan datalle suoritetaan AND-operaatio bitti bitiltä.
OR	;	PUT-lauseen matriisi siirtyy kuvaruudulla olevan kuvan päälle. Datalle suoritetaan OR-operaatio.
XOR	;	oletusarvo operaatio. Normaali XOR-operaatio kuvan ja PUT-lauseen datan kanssa.
PSET	;	GET-lauseen vastakohta. PUT-lauseen data kumoaa kuvaruudulla olevan, joten näkyviin jää vain matriisi <nimi>.
PRESET	;	PUT-lauseen matriisista otetaan negatiivikuva (vrt. valokuvaus), joka piirretään alkaen kohdasta (X,Y).

ESIM:

```
10  SCREEN 2
20  DEFINT A
30  DIM A (10,10)
40  LINE (15,15)—(35,35),15,BF
50  LINE (50,60)—(70,80),12,BF
60  GET (50,60)-(70,80),A
70  PUT (30,30),A,PSET
80  GOTO 80
```

Rivi 10 on kirjoitettava SVI-BASIC:ssä seuraavasti:

```
10  SCREEN 1
```

Rivillä 10 siirrytään tarkkuusgrafiikkamuotoon. Rivin 20 DEFINT-käsky määrittelee tulevan matriisin A kokonaislukumatriisiksi, jolle rivillä 30 varataan tilaa. Rivillä 40 piirretään suorakulmio valkoisella värillä ja rivillä 50 piirretään toinen käyttäen tummanvihreää väriä. Rivin 60 GET-käsky lukee jälkimmäisen suorakulmion matriisin A sisällöksi ja rivillä 70 se piirretään osittain ensimmäisen suorakulmion päälle. Kokeile eri operaatioita vaihtamalla rivin 70 lopussa olevaa 'PSET' sanaa edellä esitettyihin vaihtoehtoihin ja huomaa operaation vaikutus. Rivin 80 GOTO-käsky sulkee ohjelman silmukkaan, jonka voi pysäyttää painamalla CTRL+STOP.

PUT SPRITE

MUOTO: **PUT SPRITE <pinta> [,<koord>] [,<väri>] [,<num>]**

VERSIOT: **MSX: KASET. + LEVYKE**
SVI: KASET. + LEVYKE

TARKOITUS: Käskyllä asetetaan SPRITE-kuva-alkio ilmaisulla <koord> määriteltyyn kohtaan kuvaruudulla.

LISÄYKSET: <pinta> = taso, jolle SPRITE-hahmo piirretään. Käytettävissä on 32 eri tasoa, ts. jokaiselle SPRITE-hahmolle on varattu oma taso. Käyttämällä eri tasoja, saadaan aikaan syvyysvaikutelmaa. Tasot on numeroitu 0—31. Tämä on ainoa pakollinen ilmaisu käskyn yhteydessä.
<koord> = paikka kuvaruudulla, johon SPRITE-hahmo piirretään. Voidaan antaa joko absoluuttisena, jolloin muoto on seuraava:

PUT SPRITE <pinta> , (X,Y) ,...

jossa X ja Y ovat vastaavat kuvaruudun koordinaattipisteet tai ilmaisu <koord> voidaan antaa suhteellisena, jolloin muoto on seuraava:

PUT SPRITE <pinta>, STEP (X,Y),...

Nyt lopullinen SPRITE-hahmon paikka määräytyy viimeksi käsitellyn pisteen (piste, johon on viimeksi piirretty) ja STEP-lauseen koordinaattien X ja Y summana. Jos X tai Y on negatiivinen luku, suoritetaan vähennyslasku (tällöin hahmo piirtyy viimeksi piirretystä pisteestä katsoen vasemmalle ja/tai ylös).
Ilmaisun oletusarvo on piste (0,0), joka vastaa kuvaruudun vasenta yläkulmaa. Ilmaisun koordinaatit voivat saada myös arvoja, jotka menevät kuvaruudun ulkopuolelle, tosin molempien koordinaattien tulee olla välillä —32768 — +32767. Kuvaruudun ulkopuoliset koordinaatit pyöristetään aina lähimpään mahdolliseen pisteeseen.
Koordinaatti X voi saada arvot —32 — +255 ja Y-koordinaatti arvot —32 — +191. Jos Y-koordinaatti saa arvon &HD0 eli 208, kaikki tämän takana olevilla (ilmaisulla <taso> suurempi arvo, kuin SPRITE:lla, jonka Y-koordinaatti saa arvon 208) tasoilla olevat SPRITE-hahmot katoavat, kunnes Y-koordinaatti saa jonkin muun arvon. Jos Y-koordinaatti saa arvon &HD1 eli 209, katoaa kyseinen SPRITE-hahmo.
<väri> = väri, jolla SPRITE-hahmo piirretään. Kokonaisluku välillä 0—15 (kts. COLOR-käsky). Oletusarvona on tekstiväri, jota viimeksi on käytetty.
<num> = ilmaisulla määritellään piirrettävän SPRITE:n numero. Oletusarvona on ilmaisu <taso>. Ilmaisu voi olla kokonaisluku välillä 0—255, rajoituksena kuitenkin seuraava sääntö. Jos SPRITE-hahmojen kooksi on SCREEN-käskyllä määritelty 0 tai 1, on käytössä numerot 0—255 ja jos SPRITE-hahmojen kooksi on määritelty 2 tai 3, on käytössä numerot 0—63. Kerrallaan voidaan kuvaruudussa ohjata 32:ta SPRITE-hahmoa, määritellä SPRITE$-käskyllä voidaan kaikki 255.
Kts. SPRITE$- ja SCREEN käskyt.

ESIM:

```
10  SCREEN 2
20  FOR A% = 1 TO 8
30  READ A$
40  B$ = B$ + CHR$ (VAL ("&B" + A$))
50  NEXT
60  DATA 00000000
70  DATA 00111100
80  DATA 00100000
90  DATA 00111100
100 DATA 00000100
```

```
110 DATA 00000100
120 DATA 00111100
130 DATA 00000000
140 REM
150 SPRITE$(0)=B$
160 SPRITE$(1)=B$
170 SPRITE$(2)=B$
180 REM
190 PUT SPRITE 0,(128,100),15
200 PUT SPRITE 2,(132,104),7
210 PUT SPRITE 1,(136,108),13
220 GOTO 220
```

Rivi 10 on SVI-BASIC:ssä kirjoitettava seuraavasti:

```
10 SCREEN 1
```

Rivillä 10 siirrytään tarkkuusgrafiikkamuotoon. Rivit 20—50 muodostavat FOR-NEXT -silmukan, jonka aikana SPRITE-hahmo saa muodon. Rivillä 30 luetaan binaaridata, joka rivillä 40 muunnetaan VAL- ja CHR$-funktioiden avulla merkkijonomuuttujan B sisällöksi. Rivin 50 NEXT-käsky sulkee FOR-silmukan. Rivit 60—130 sisältävät SPRITE-hahmon muodon binaarimuodossa (1- piste sytytetään, 0- pistettä ei sytytetä). Riveillä 140 ja 180 olevat REM-käskyt eivät vaikuta ohjelman suoritukseen. Riveillä 150—170 määritellään SPRITE:lle numerot 1, 1 ja 2 muuttujan B$ sisältämä data. Rivillä 190 piirretään SPRITE-hahmo numero 0 tasolle 0, pisteestä (128,100) alkaen, valkoisella värillä. Rivin 200 PUT SPRITE -käsky piirtää SPRITE-hahmon numero 2 pinnalle 2, värillä 7. Rivillä 210 piirretään SPRITE-hahmo pisteestä (136,108) alkaen. Kuvaruudulta voi selvästi havaita SPRITE-hahmojen olevan eri tasoilla (syvyysvaikutelma). Muuta rivillä 210 olevat pisteen koordinaatit seuraaviksi: (138,208). Nyt sekä tasolla 1 että 2 olevat SPRITE-hahmot katoavat kuvaruudulta ja vain rivin 200 SPRITE-hahmo jää näkyviin (johtuen Y-koordinaatin 208 aiheuttamasta tilanteesta.) Vaihda koordinaateiksi (136,209) ja huomaa, kuinka vain SPRITE-hahmo numero 1 katoaa.

SCREEN

Koska SCREEN-käsky eroaa MSX- ja SVI-BASIC:ssä huomattavasti, käsitellään käsky molempien BASIC-versioiden kannalta erikseen.

MSX-BASIC

MUOTO: **SCREEN [<moodi>] [,<sprite>] [,<key>] [,<cass>] [,<printer>]**

VERSIOT: **MSX: KASET. + LEVYKE**

TARKOITUS: Käskyllä vaihdetaan näyttömoodia, SPRITE-kuva-alkioiden kokoa, näppäinten "klik"-ääni, kasettitallentimen nopeutta tai kirjoittimen moodia.

LISÄYKSET: <moodi> = ilmaisu on kokonaisluku väliltä 0—3, jolla vaihdetaan näytön tilaa seuraavasti:

MOODI		TOIMINTA
0	;	40 saraketta * 24 riviä (maks.) sisältävä tekstimuoto. Komentotila, johon tullaan, kun koneeseen kytketään virta (POWER ON). Oletusarvona 39 saraketta * 23 riviä (rivi 24 on varattu funktionäppäimille, mutta voidaan vapauttaa KEY OFF -käskyllä).
1	:	32 saraketta * 24 riviä (maks.) sisältävä levennetty tekstimoodi (jokainen merkki varaa 1 ja 1/4 merkin verran tilaa). Ohjelmarivien editointi on sallittua.
2	;	tarkkuusgrafiikkatila, jossa kuvaruutu on jaettu 256*192 pisteeseen. Käytettävissä on 16 väriä, tosin yksi kuva-alkio (8*8 pistettä) voi pitää vain yhtä väriä kerrallaan.
3	;	lohkografiikkatila, jossa kuvaruutu on jaettu 64*48 pisteeseen. Kaikki 16 väriä ovat käytettävissä. SPRITE-hahmot voidaan piirtää yhden pikselin (pisteen) tarkkuudella.

<sprite> = ilmaisulla määritellään SPRITE-kuva-alkion koko seuraavasti:

MUOTO		SPRITE-ALKION KOKO
0	;	8*8 pistettä. Muodostetaan kahdeksasta 8-bittisestä datasta. Suurentamaton.

1 ; 16*16 pistettä. Muodostetaan kahdeksasta 8-bittisestä datasta, jotka BASIC nelinkertaistaa. Yksi alkio varaa siis neljän kuva-alkion verran tilaa kuvaruudulta.

2 ; 16*16 pistettä. Muodostetaan 32:sta 8-bittisestä datasta seuraavasti:

I	III
II	IV

I = kahdeksan ensimmäistä, II = kahdeksan seuraavaa dataa, jne.

3 ; 32*32 pistettä. Muodostetaan 32:sta 8-bittisestä datasta, jotka BASIC nelikertaistaa. Vrt. SPRITE-koko kaksi (2).

<key> = ilmaisulla asetetaan näppäimiä painettaessa kuuluva "klik"-ääni kuuluviin seuraavasti:

0 ; eliminoi äänen.
ei nolla ; asettaa äänen kuuluviin (ei negat. luku).

<cass> = ilmaisulla vaihdetaan kasettinauhurin tiedontallennusnopeutta. Nopeuksia on kaksi:

1 ; asettaa nopeudeksi 1200 baudia (bittiä/sekunti).
2 ; asettaa nopeudeksi 2400 baudia.

Muilla arvoilla tulostuu virheilmoitus "Illegal function call". Tallennusnopeus voidaan asettaa myös CSAVE-käskyllä, joten käsky on tarkoitettu lähinnä konekieliohjelmien tallennuksessa. Luettaessa ohjelmia nauhalta EI nopeutta voi muuttaa, vaan luku tapahtuu automaattisesti nopeudella, jolla ohjelma on taltioitu.
<printer> = ilmaisu kertoo, onko käytössä MSX- vai tavallinen kirjoitin. MSX-kirjoitin tukee grafiikkamerkkejä, joita standardi Centronic-kirjoitin ei tunne (ts. MSX-kirjoitin osaa tulostaa MSX-grafiikkamerkkejä). Ilmaisulla ohjataan seuraavasti:
0 ; käytössä on MSX-kirjoitin (oletusarvo).
ei nolla ; käytössä on standardi Centoronic-kirjoitin (grafiikkamerkit muunnetaan välilyönneiksi eli tyhjiksi).

SPECTRAVIDEO

MUOTO: **SCREEN [<moodi>] [,<sprite>]**

VERSIOT: **SVI: KASET. + LEVYKE**

TARKOITUS: Käskyllä vaihdetaan näyttömoodia, SPRITE-kuva-alkioiden kokoa tai asetetaan/poistetaan funktionäppäinten merkitykset näytöstä.

LISÄYKSET: <moodi> = ilmaisu on kokonaisluku väliltä 0—2, jolla vaihdetaan näytön tilaa seuraavasti:

MOODI	TOIMINTA
0	40 saraketta * 24 riviä (maks.) sisältävä tekstimuoto. Komentotila, johon tullaan, kun koneeseen kytketään virta (POWER ON). Oletusarvona 39 saraketta * 23 riviä (rivi 24 on varattu funktionäppäimille, mutta voidaan vapauttaa KEY OFF -käskyllä).
1	tarkkuusgrafiikkatila, jossa kuvaruutu on jaettu 256*192 pisteeseen. Käytettävissä on 16 väriä, tosin yksi kuva-alkio (8*8 pistettä) voi pitää vain yhtä väriä kerrallaan.
2	lohkografiikkatila, jossa kuvaruutu on jaettu 64*48 pisteeseen. Kaikki 16 väriä ovat käytettävissä. SPRITE-hahmot voidaan piirtää yhden pikselin (pisteen) tarkkuudella.

<sprite> = ilmaisulla määritellään SPRITE-kuva-alkion koko seuraavasti:

MUOTO	SPRITE-ALKION KOKO
0	8*8 pistettä. Muodostetaan kahdeksasta 8-bittisestä datasta. Suurentamaton.
1	16*16 pistettä. Muodostetaan kahdeksasta 8-bittisestä datasta, jotka BASIC nelinkertaistaa. Yksi alkio varaa siis neljän kuva-alkion verran tilaa kuvaruudulta.
2	16*16 pistettä. Muodostetaan 32:sta 8-bittisestä datasta seuraavasti:

I	III
II	IV

I = kahdeksan ensimmäistä, II = kahdeksan seuraavaa dataa, jne.

3 ; 32*32 pistettä. Muodostetaan 32:sta 8-bittisestä datasta, jotka BASIC nelikertaistaa. Vrt. SPRITE-koko kaksi (2).

SCREEN-käskyllä voidaan SVI-BASIC:ssä poistaa funktionäppäinten merkitykset kuvaruudun alareunasta tai asettaa ne uudelleen näkyviin (MSX-BASIC:ssä käsky KEY ON/OFF). Muoto on seuraava:

SCREEN ,0 ; poistaa merkitykset kuvaruudusta.
SCREEN ,1 ; asettaa merkitykset näkyviin.

Huomaa käskyn SCREEN ,0 ero eri moodien välillä.
SCREEN-käsky sisältää automaattisen CLS-käskyn ja palauttaa näppäinten alkuperäiset merkitykset, mikäli niitä ohjelmallisesti on muutettu.

SPRITE $

MUOTO: **SPRITE$(<num>) = <string>**
VERSIOT: **MSC: KASET. + LEVYKE**
SVI: KASET. + LEVYKE

TARKOITUS: Muuttujalla määritellään SPRITE-alkio.

LISÄYKSET: <num> = kokonaisluku väliltä 0—255, kun SPRITE-alkioiden koko on 0 tai 1 (kts. SCREEN-käsky) tai väliltä 0—63, kun SPRITE-alkioiden koko on 2 tai 3.
Kerrallaan voidaan ohjata kuvaruudulla 32 eri SPRITE-alkiota, mutta määriteltynä voi olla kaikki 256 kpl (koko on 0 tai 1).
Jokainen SPRITE-alkio varaa 32 tavua muistia, joten jos alkion määrittely sisältää esim. kahdeksan 8-bittistä dataa (SPRITE-koko 0 tai 1), täyttää BASIC loput 24 muistipaikkaa CHR$(0)-koodilla
<string> = ilmaisulla määritellään SPRITE-alkion muoto. Ilmaisu voi olla merkkijonomuuttuja tai CHR$-koodeja sisältävä lause

ESIM:

```
10  SCREEN 2,3
20  A$= CHR$(&B10000001) + CHR$(&B01000010) + CHR$(&H24) + CHR$
    (&H18)
30  A$= A$ + CHR$(&H18) + CHR$(&H24) + CHR$(42) + CHR$(&H81)
40  SPRITE$(1) = A$
50  SPRITE$(2) = A$ + A$
60  SPRITE$(3) = A$ + A$ + A$
70  SPRITE$(4) = A$ + A$ + A$ + A$
80  PUT SPRITE 1,(50,80),15
90  PUT SPRITE 2,(100,80),13
100 PUT SPRITE 3,(150,80),8
110 PUT SPRITE 4,(200,80),2
120 GOTO 120
```

Rivi 10 on SVI-BASIC:ssä kirjoitettava seuraavasti:

```
10  SCREEN 1,3
```

Rivillä 10 siirrytään tarkkuusgrafiikkamuotoon ja määritellään suurimmaksi SPRITE-alkion kooksi 32*32 pistettä. Rivillä 20 määritellään merkkijonomuuttujalle A neljä ensimmäistä SPRITE-alkion dataa, joista kaksi ensimmäistä on binaarimuodossa ja kaksi seuraavaa heksadesimaalimuodossa. Rivillä 30 lisätään muuttujaan A$ neljä data-alkiota, jotka kaikki ovat heksamuodossa. Rivillä 40 määritellään SPRITE-alkio numero yksi (1), joka saa muuttujan A$ sisältämän tiedon. Rivillä 50 määritellään SPRITE-alkio numero kaksi (2), joka saa kaksi kertaa samat tiedot, kuin SPRITE-alkio yksi. Riveillä 60 ja 70 saa SPRITE-alkiot 2 ja 3 vastaavasti kaksinkertaisen ja kolminkertaisen määrän tietoa (vrt. tulostukseen ohjelman ajon yhteydessä). Rivillä 80—110 tulostetaan SPRITE-alkiot 1—4, kukin omalla värillään. Rivin 120 GOTO-käsky sulkee ohjelman suorituksen silmukkaan, jonka voi pysäyttää painamalla CTRL+STOP.

STICK

MUOTO: **STICK (<num>)**

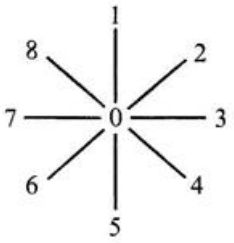
VERSIOT: **MSX: KASET. + LEVYKE**
SVI: KASET. + LEVYKE

TARKOITUS: Funktio palauttaa peliohjaimen <num> asennon.

LISÄYKSET: <num> = ilmaisu on jokin seuraavista:

NUM		TARKOITTAA
0	;	kursorinohjainnäppäimiä (MSX ja SVI-328) tai kiinteää peliohjainta (SVI-318).
1	;	peliohjainta numero yksi (MSX ja SVI).
2	;	peliohjainta numero kaksi (MSX ja SVI).

Funktio on muotoa <muuttuja> = STICK (<num>), jossa ilmaisu <muuttuja> saa jonkin seuraavista arvoista:



Jos peliohjainta (kursorinohjainnäppäimiä) ei käännetä mihinkään suuntaan, palautetaan siis nolla (0).

ESIM:

```
10  SCREEN 2
20  X%=128 : Y%=96
30  PSET (X%,Y%),15
40  A%=STICK(0)
50  IF A%=0 THEN GOTO 30
60  PRESET (X%,Y%)
70  ON A% GOSUB 80,90,100,110,120,130,140,150
80  Y%=Y%—1 : RETURN 30
90  Y%=Y%—1 : X%=X%+1 : RETURN 30
100 X%=X%+1 : RETURN 30
110 Y%=Y%+1 : X%=X%+1 : RETURN 30
120 Y%=Y%+1 : RETURN 30
130 Y%=Y%+1 : X%=X%—1 : RETURN 30
140 X%=X%—1 : RETURN 30
150 Y%=Y%—1 : X%=X%—1 : RETURN 30
```

Rivi 10 on SVI-BASIC:ssä kirjoitettava seuraavasti:

```
10  SCREEN 1
```

Rivillä 10 siirrytään tarkkuusgrafiikkamuotoon. Rivillä 20 annetaan aloituspisteen koordinaatit (128,96) keskelle kuvaruutua. Rivillä 30 piirretään piste muuttujien X ja Y määräämään kohtaan. Rivillä 40 tutkitaan STICK-funktion avulla painetaanko kursorinohjainnäppäimiä (muuttamalla luku nolla esim. yhdeksi, voidaan pistettä ohjata peliohjaimella). Rivillä 50 määrätään, että jos kursorinäppäimiä ei ole painettu, palataan riville 30 asettamaan piste uudelleen. Riville 60 siirrytään, jos jotain kursorinäppäintä on painettu ja silloin poistetaan vanha piste. Rivillä 70 suoritetaan päättely painetun näppäimen mukaan. Muuttuja A% sisältää tiedon, mitä näppäintä on painettu. Rivin 70 ON GOSUB -lauseella suoritetaan hyppy muuttujan A% arvoa vastaavalle riville. Rivit 80—150 sisältävät ohjeet pisteen siirtämisestä seuraavaan paikkaan ja lopuksi paluun riville 30 piirtämään pistettä uuteen pisteeseen. Jos pistettä on ohjattu kursorinohjainnäppäimillä esim. eteenpäin, on muuttujan A% arvo yksi (1), jolloin riviltä 70 siirrytään riville 80. Rivillä 80 vähennetään Y-koordinaattia yhdellä ja palataan riville 30 piirtämään pistettä uuteen paikkaan, jne.

STRIG

MUOTO: **STRIG (X)**

VERSIOT: **MSX: KASET. + LEVYKE**
SVI: KASET. + LEVYKE

TARKOITUS: Funktio palauttaa peliohjaimen FIRE-näppäimen tilan.

LISÄYKSET: Ilmaisulla X osoitetaan joko välilyöntinäppäintä tai peliohjainta 1 tai 2. X:n arvo nolla (0) tarkoittaa välilyöntiä, arvo yksi (1) peliohjainta yksi ja X:n arvo kaksi (2) peliohjainta kaksi.

Funktio palauttaa arvon nolla (0) eli EPÄTOSI, jos vastaavaa näppäintä EI ole painettu ja arvon 1- eli TOSI, jos näppäintä on painettu.
Funktiota käytetään yleensä IF-lauseessa.

ESIM:

```
10  CLS
20  LOCATE 8,10
30  PRINT "PAINA VÄLILYÖNTINÄPPÄINTÄ,"
40  LOCATE 8,11
50  PRINT " JA KUULET BEEP-ÄÄNEN."
60  IF STRIG(0) THEN 70 ELSE 60
70  BEEP
80  END
```

Rivillä 10 tyhjennetään kuvaruutu CLS-käskyllä. Rivin 20 LOCATE-käsky kohdistaa kursorin riville 10, sarakkeeseen 8, josta seuraavasta sarakkeesta alkaen tulostaa rivin 30 PRINT-lause ohjeita. Rivillä 40 kohdistetaan kursori seuraavalle riville samaan kohtaan. Rivillä 50 tulostetaan loput ohjeet. Rivin 60 IF-lause tutkii, onko välilyöntinäppäintä painettu (muuttamalla suluissa oleva arvo nolla esim. luvuksi yksi, voidaan ohjaus antaa peliohjaimella). Jos välilyöntiä ei ole painettu, pysyy suoritus rivillä 60 (IF-lauseen totuusarvo on EPÄTOSI) ja jos näppäintä on painettu, siirrytään riville 70, jolla annetaan "piip"-ääni. Rivin 80 END-käsky päättää ohjelman.

VDP

MUOTO: VDP (X)

VERSIOT: MSX: KASET. + LEVYKE
SVI: ei käytössä

TARKOITUS: Funktiolla luetaan/kirjoitetaan suoraan VDP:n (Video Display Processor eli videoprosessori) rekistereihin.

LISÄYKSET: Ilmaisulla X annetaan rekisteri, johon luku/kirjoitus kohdistuu. VDP sisältää kahdeksan (8) rekisteriä, jotka ovat seuraavat:

REKISTERI		TOIMINTA
0—7	;	Write Only -rekisteri (vain kirjoitus rekisteriin on mahdollista). Kirjoitettava luku on oltava välillä 0—255.
8	;	Read Only -rekisteri (vain rekisterin luku on mahdollista). Palauttaa luvun väliltä 0—255. On VDP:n tilarekisteri (Status Register).

VDP:n ohjauksesta löytyy SPECTRAVIDEO:n käyttäjille lisää tietoa kirjasta SERVICE & TECHNICAL MANUAL.

ESIM:

```
PRINT VDP (8) <ENTER>
  XXX
Ok
```

Esimerkkilause lukee VDP:n tilarekisterin ja palauttaa sen arvon kuvaruudulle (XXX).

VPEEK

MUOTO: VPEEK (X)

VERSIOT: MSX: KASET. + LEVYKE
SVI: KASET. + LEVYKE

TARKOITUS: Funktio palauttaa tiedon videomuistin osoitteesta X.

LISÄYKSET: Sekä MSX- että SPECTRAVIDEO-mikrotietokoneissa on RAM-muisti jaettu erikseen työ- ja videomuistiin. Videomuistin koko on 16 kilotavua (16384 muistipaikkaa, jossa kussakin yksi merkki), joka on siis erotettu varsinaisesta työmuistista. Tällä tavoin on BASIC:lle vapaan (käyttäjän ohjelmille vapaa RAM-muistitila) muistin määrä saatu mahdollisimman suureksi.
Videomuisti sisältää tiedon siitä, mitä kuvaruudulla näkyy kullakin hetkellä. Tiedot videomuistissa on tallennettu lohkoihin erikseen teksti- ja grafiikkatilaa varten. Esimerkiksi tekstitilaa vastaavat osoitteet ovat 0—959, joista osoite 0000 vastaa kuvaruudun vasenta yläkulmaa (WIDTH 40), osoite 0039 oikeaa yläkulmaa, osoite 0040 saraketta yksi rivillä 2 ja osoite 00959 rivin 24 oikeaa kulmaa (kuvaruudun oikea alakulma).

Osoitteen X tulee olla välillä 0—16383. Arvo, jonka funktio palauttaa, on välillä 0—255. SVI-BASIC:ssä tulee palautettuun arvoon lisätä luku 32, jotta saataisiin vastaava ASCII-koodi.
VPEEK-funktio on funktion VPOKE käänteisfunktio.
Kts. VPOKE-funktio ja vertaa PEEK-funktioon.

ESIM:

```
10  WITH 40
20  FOR A% = 920 TO 959
30  PRINT VPEEK(A%);
40  NEXT
```

Rivi 30 on muutettava SVI-BASIC:ssä seuraavaksi:

```
30  PRINT VPEEK (A%) + 32;
```

Rivillä 10 muunnetaan tulostuksen leveydeksi 40 saraketta/rivi, jotta funktionäppäinten merkitykset näkyy samanlaisina, kuin ne näkyvät kuvaruudun alareunassa. Rivit 20—40 muodostavat FOR-NEXT -silmukan, joka siis tulostaa funktionäppäinten merkitykset kuvaruutuun. Rivin 30 VPEEK-käsky lukee videomuistista osoitteesta A% tiedon, joka on ASCII-koodi kuvaruudun riviltä 24. SVI BASIC:ssä on palautettavaan lukuun lisättävä luku 32, siksi muutos riville 30.

VPOKE

MUOTO: VPOKE <addr>,<data>

VERSIOT: MSX: KASET. + LEVYKE
SVI: KASET. + LEVYKE

TARKOITUS: Funktio kirjoittaa videomuistiin osoitteeseen <addr> tiedon <data>.

LISÄYKSET: <addr> = osoite, johon tieto talletetaan. Ilmaisu voi olla välillä 0—16383 ja se voidaan antaa desimaali-, heksadesimaali-, oktaali- tai binaarilukuna. Ilmaisu voi olla myös muuttuja.
<data> = tieto, joka talletetaan. Ilmaisu voi olla kokonaisluku välillä 0—255 tai voidaan antaa muuttujana.
On huomattava, että muutamat käskyt palauttavat videomuistiin tehdyt muutokset. Näitä käskyjä ovat mm. CLS, SCREEN, WIDTH ja SHIFT-näppäimen painallus. Kokeile näiden käskyjen vaikutusta alla olevan esimerkkiohjelman ajon jälkeen.
SVI-BASIC:ssä on VPOKE-käskyn ilmaisusta <data> vähennettävä luku 32 ennen tiedon tallentamista tai viimeistään VPOKE-käskyn yhteydessä, jos tieto halutaan tallentaa ASCII-koodina.
Vrt. POKE-käskyyn ja katso VPEEK-funktio.

ESIM:

```
10  FOR A% = 920 TO 959
20  VPOKE A%,42
30  NEXT
```

Rivi 20 on kirjoitettava SVI-BASIC:ssä seuraavasti:

```
20  VPOKE A%,10
```

Ohjelma tulostaa rivin 24 täyteen tähtiä (*), ASCII-koodi 42. Rivit 10—30 muodostavat FOR-NEXT -silmukan, jossa rivillä 20 kirjoitetaan funktionäppäinten tilalle tähtiä. Rivin 30 NEXT-käsky sulkee silmukan. Huomaa kuitenkin, että muutos ei ole pysyvä, vaan jopa SHIFT-näppäimen yksi painallus riittää palauttamaan alkuperäisen tekstin.

WIDTH

MUOTO: WIDTH <nro>

VERSIOT: MSX: KASET. + LEVYKE
SVI: KASET. + LEVYKE

TARKOITUS: Funktiolla määritellään rivin leveys kuvaruudulla.

LISÄYKSET: <nro> = luku, jolla rivin leveys määritellään.

SVI-BASIC:ssä ilmaisu <nro> voi olla joko 39, 40 tai 80 (tällöin käytössä tulee olla 80 merkkiä/rivi laajennusyksikkö ja DISK BASIC). MSX-BASIC:ssä voi <nro> olla väliltä 1—40 (normaalissa tekstitilassa) tai väliltä 1—32 (laajennettu tekstitila).
Käskyn käyttö ei vaikuta kirjoittimelle tapahtuvaan tulostukseen.

ESIM:

```
WIDTH 28    ; toimii vain MSX-BASIC:ssä. Vaihtaa rivin leveydeksi kuvaruudulla 28 merkkiä/rivi.
```

TIEDOSTOKÄSKYT

Tiedostot jaetaan kahteen tyyppiin: sarjasaanti- ja hajasaantitiedostoihin. Sarjasaantitiedostoihin kirjoitus ja luku tapahtuu järjestyksessä, alkaen ensimmäisestä tietueesta viimeiseen tietueeseen. Sarjasaantitiedosto on luettava tai kirjoitettava aina alusta loppuun asti, vaikka tarvittaisiin vain yhtä tietuetta. Tämä ominaisuus tekee käytön hitaaksi, mutta yksinkertaiseksi. Hajasaantitiedostoissa voidaan viitata mihin tahansa tietueeseen mielivaltaisessa järjestyksessä. Hajasaantitiedostojen huonona puolena voidaan pitää mm. sitä, että luvut on muunnettava merkkijonomuotoon ennen niiden tallentamista (kts. CVI-, CVS- ja CVD-funktiot) ja suhteellisen vaikea tapa, jolla tiedostoja käsitellään (muistipuskurit, joiden välityksellä tietoja siirretään). Hajasaantitiedostot ovat kuitenkin huomattavasti sarjasaantitiedostoja nopeampia, kun käsiteltävää tietoa on paljon.
Jos käytössä on datanauhuri (tai normaali kasettitallennin), voidaan käyttää vain sarjasaantitiedostoja ja jos käytössä on levykeasema, voidaan käyttää molempia tiedostotyyppejä.
Tiedostonumeroiden edellä voidaan käyttää #-merkkiä.

ATTR $

MUOTO: **ATTR$ ("<nro>:<fname>")**

VERSIOT: **MSX: ei käytössä**
SVI: LEVYKE

TARKOITUS: Funktio palauttaa voimassa olevat ehdot ohjelmalle/tiedostolle <fname>.

LISÄYKSET: <nro> = levykeaseman numero, jolla olevaa ohjelmaa tutkitaan. Ilmaisun arvo on yksi (1), jos ohjelma on levykeasemassa yksi ja kaksi (2), jos ohjelma on asemassa kaksi.
<fname> = tiedosto/ohjelma, jonka tila halutaan tietää.
Ohjelmalle voidaan antaa ehto SET-käskyllä. Ehdot ovat seuraavat:

P ; antaa ohjelmalle kirjoitussuojauksen.
R ; ohjelma tarkistetaan joka kerta, kun se luetaan (esim. ASCII-tiedostot).

ESIM:
```
PRINT ATTR$ ("1:ipl") <ENTER>
R
Ok
```

Ohjelma nimeltä "ipl" tarkistetaan joka luvun yhteydessä.

CLOSE

MUOTO: **CLOSE [#] [<fnum>] [,<fnum2> ,...]**
VERSIOT: **MSX: KASET. + LEVYKE**
SVI: KASET. + LEVYKE

TARKOITUS: Käsky sulkee avoinna olevan tiedoston tai tiedostot.

LISÄYKSET: Sulkemisen jälkeen on suljettujen tiedostojen käytössä olleet tiedostonumerot ja tiedostopuskuri uudelleen käytettävissä. Vain avoinna olevat tiedostot voidaan sulkea.
Myös käskyt END, CLEAR, MAXFILES ja NEW sulkevat tiedostot ilman CLOSE-käskyä. STOP-käsky EI sulje avoimia tiedostoja.

ESIM:
```
10  OPEN "CAS:KOE" FOR OUTPUT AS #1
20  PRINT #1,"TEKSTIÄ JA NUMEROITA 123456"
30  CLOSE #1
```

Rivillä 10 avataan sarjasaantitiedosto nimellä "KOE" kirjoitusta varten numerolla yksi (#1). Rivillä 20 kirjoitetaan tiedostoon yksi lainausmerkkien sisällä oleva teksti. Rivin 30 CLOSE-käsky sulkee tiedoston yksi. Huomioi, että #-merkki ei ole pakollinen OPEN- ja CLOSE-käskyjen yhteydessä.

COPY

MUOTO: **MSX: COPY " [<drive1>:] file1" TO " [<drive2>:] file2"**
SVI: COPY <nro1> FROM "<nro2>:<fname>"

VERSIOT: **MSX: LEVYKE**
SVI: LEVYKE

TARKOITUS: Käsky kopioi ohjelman levykeasemalta toiselle.

LISÄYKSET:

SVI-BASIC

<nro1> = levykeasema, jolle kopioitava ohjelma taltioidaan. Joko yksi (1) tai kaksi (2).
<nro2> = levykeasema, jolta ohjelma kopioidaan. Joko yksi (1) tai kaksi (2).
<fname> = tiedoston/ohjelman nimi, joka kopioidaan. Tulee vastata täydellisesti levykkeellä olevan ohjelman nimeä.
Käytössä tulee olla kaksi levykeasemaa.

MSX-BASIC

Käskyllä voidaan kopioida yksi tai useampia tiedostoja/ohjelmia. Jos kopioidaan useampi, kuin yksi ohjelma, tulee käyttää merkkejä ? ja *. Kysymysmerkillä (?) voidaan tiedostonimestä korvata mikä tahansa merkki asettamalla halutun merkin kohdalle ?-merkki. Näin kopioituu kaikki tiedostot, joiden nimi täyttää määritellyn ehdon. Tähdellä (*) voidaan korvata kaikki merkit, alkaen tähdestä. Esim. käsky COPY "A: *.*" TO "B: *.*" kopioi kaikki levykeasemalla A olevat ohjelmat levykeasemaan B.
Levykeasemia voi olla yksi tai kaksi. Jos käytössä on yksi asema, pyytää kone välillä vaihtamaan diskettiä seuraavalla ilmoituksella:

Insert diskette for drive X:
and strike a key when ready

Kirjain X kuvaa levykeasemaa, johon luku/kirjoitus seuraavaksi kohdistuu. Levykeaseman voidaan ajatella olevan levyke itse, jos käytössä on vain yksi asema. Levykeaseman oletusarvona on asema A (yksi). Kun oikea levyke on asemassa ja lukitusvipu suljettuna, painetaan jotain näppäintä. Jos lukitusvipu ei ole suljettuna, tulostuu virheilmoitus "Disk offline" ja käsky on annettava uudestaan.
Kopioitavan ohjelman nimi voidaan muuttaa, antamalla nimi <file2> eri nimeksi, kuin kopioitava ohjelma (<file1>).
<drive1> = levykeasema, jolta ohjelma/ohjelmat kopioidaan. Merkitään kirjaimella A (asema yksi, joka on myös oletusarvo alkutilassa) tai kirjaimella B (asema kaksi).
<drive2> = levykeasema, jolle kopioitava ohjelma taltioidaan. Jos ilmaisua ei anneta, kopioidaan ohjelma oletusarvoasemaan (asema A).
<file1> = ohjelma/ohjelmat, jotka kopioidaan. Jos kopioidaan useampi, kuin yksi ohjelma, käytetään merkkejä ? ja *. Huom! erikoismerkkejä ? ja * voidaan käyttää vain MSX DISK BASIC:ssä (MSX DOS).
<file2> = nimi, jolla kopioitava ohjelma levykkeelle taltioidaan. Jos ilmaisua ei anneta, taltioituu ohjelma alkuperäisellä nimellä.
Jos kopioitava ohjelma on pitkä ja käytössä on yksi levykeasema, suoritetaan kopiointi osissa ja näin joudutaan levykkeitä vaihtamaan, kunnes kopiointi on loppuun suoritettu. Levykkeen vaihdosta ilmoittaa kone tulostamalla "Insert diskette...".
Viimeksi käytössä ollut levykeasema jää aina oletusarvoksi, joten esim. käskyn FILES "B: *.*" jälkeen jää oletusarvoasemaksi asema B (kaksi). Jos tämän jälkeen annat esimerkiksi käskyn FILES, pyytää kone asettamaan levykkeen asemaan A (yksi).
MSX DISK BASIC muuttaa ohjelmien nimet isoiksi kirjaimiksi ennen levykkeelle taltiointia.

ESIM:

KÄSKY	**KOPIOI**
COPY 1 FROM "2:plist"	; asemasta kaksi asemaan yksi ohjelman nimeltä "plist". Toimii vain SVI DISK BASIC:ssä.
COPY "B:*.BAS" TO "A:*.ASC"	; asemasta B (2) kaikki BAS-loppuiset ohjelmat asemaan A, vaihtaen ohjelmat ASC-loppuisiksi. Toimii vain MSX DISK BASIC:ssä.
COPY "test" TO "rest"	; ohjelman (oletusarvoasemasta) nimeltä "TEST" oletusarvoasemaan nimellä "REST". Ohjelman nimi luetaan ja taltioidaan isoilla kirjaimilla. Toimii vain MSX BASIC:ssä.

CVI, CVS, CVD

MUOTO:

CVI (<string>)
CVS (<string>)
CVD (<string>)

VERSIOT:

MSX: LEVYKE
SVI: LEVYKE

TARKOITUS: Funktioilla muunnetaan levykkeeltä (random access eli hajasaantitiedostosta) luetut merkkijonot takaisin luvuksi.

LISÄYKSET: Ennen kuin luku voidaan taltioida levykkeelle hajasaantitiedostoon, on luku muunnettava merkkijonoksi. CVI-, CVS- ja CVD-funktiot muuntavat merkkijonot takaisin luvuiksi seuraavasti:
CVI-funktio muuntaa kahden (2) tavun merkkijonon kokonaisluvuksi.
CVS-funktio muuntaa neljän (4) tavun merkkijonon yksinkertaisen tarkkuuden luvuksi.
CVD-funktio muuntaa kahdeksan (8) tavun merkkijonon kaksinkertaisen tarkkuuden luvuksi.

ESIM:

```
....
210 FIELD #1, 2 AS YEAR$
220 GET #1,10
230 Y%=CVI(YEAR$)
....
```

Rivillä 2100 varataan FIELD-puskurista (tiedostosta numero yksi) tilaa kaksi merkkiä merkkijonomuuttujalle YEAR$. Rivillä 220 luetaan tietue kymmenen. Rivillä 230 muunnetaan luettu merkkijono YEAR$ kokonaisluvuksi muuttujan Y% sisällöksi.

DSKF

MUOTO: **DSKF (X)**

VERSIOT: **MSX: LEVYKE**
SVI: LEVYKE

TARKOITUS: Funktio palauttaa vapaiden urien määrän (SVI DISK BASIC) tai vapaiden 256-tavuisten tietueiden määrän (MSX).

LISÄYKSET: Ilmaisu X tarkoittaa levykeasemaa, jolta käyttämättömien urien määrä lasketaan. Arvo yksi (1) vastaa asemaa yksi (SVI DISK BASIC), arvo kaksi (2) asemaa kaksi (SVI) tai asemaa B (MSX) ja arvo nolla (0) oletusarvoasemaa (MSX).

ESIM:

```
PRINT DSKF(1) <ENTER>
 XXX
Ok
```

Funktio palauttaa määrän XXX.

DSKI $

MUOTO: **DSKI$ (<drno>,<track>,<sect>)**

VERSIOT: **MSX: LEVYKE**
SVI: LEVYKE

TARKOITUS: Funktio palauttaa määritellyn sektorin sisällön kuvaruudulle (SVI) tai RAM-muistiin muistipaikkojen &HF351 ja &HF352 osoittamasta osoitteesta alkaen.

LISÄYKSET: <drno> = levykeasema, jolta sektori luetaan. Voi olla yksi (1) tai kaksi (2) MSX- ja SVI-BASIC:ssä tai nolla (0) vastaamassa oletusarvoasemaa MSX DISK BASIC:ssä.
<track> = ura, jolta sektori luetaan. Ilmaisu on välillä 0—39.
<sect> = sektori, joka luetaan. Voi olla välillä 1—17.
Funktio palauttaa sektorin 255 ensimmäistä tavua.
MSX DISK BASIC:ssä funktio palauttaa kuvaruutuun nollan (0) ja tieto talletetaan muistipaikkojen &HF351 ja &HF352 osoittamaan muistialueeseen.
Funktio on DSKO$-funktion käänteisfunktio.

ESIM:

```
10  CLEAR 255
20  PRINT DSKI$ (1,10,15)
```

Ohjelma lukee levyasemasta yksi (1), uralta 10 sektorin 15 ja tulostaa sen kuvaruudulle (vrt. rivi 20). Rivillä 10 varataan muistista tilaa palautettavalle sektorille, sillä muistissa on perustilanteessa 200 merkkiä tilaa ja funktio palauttaa 255:n merkin pituisen merkkijonon.

DSKO $

MUOTO: **MSX: DSKO$ <drno>,<locsect>**
SVI: DSKO$ <drno>,<track>,<sect>

VERSIOT: **MSX: LEVYKE**
SVI: LEVYKE

TARKOITUS: Käskyllä kirjoitetaan tietoa määrätylle sektorille.

LISÄYKSET: <drno> = levykeasema, jolle tieto talletetaan. Arvo nolla (0) vastaa oletusarvoasemaa (toimii vain MSX-BASIC:ssä), arvo yksi (1) asemaa yksi ja arvo kaksi (2) asemaa kaksi.
<locsect> = looginen sektorin numero, jolle tieto kirjoitetaan muistipaikkojen &HF351 ja &HF352 osoittamista paikasta alkaen (RAM-muistia). Ilmaisun arvoa ei testata, joten virheellinenkin arvo hyväksytään. Käytössä vain MSX-versiossa.
Huomioi, että muistipaikkojen &HF351 ja &HF352 osoittama alue nollautuu mm. käskyjen CLOSE, FILES, OPEN, jne. yhteydessä.
<track> = ura, jolla sektori sijaitsee. Kokonaisluku välillä 0—39.
<sect> = sektori, jolle tieto talletetaan. Kokonaisluku välillä 1—17.
Talletettavan merkkijonon maksimi pituus on 255 merkkiä. Jos merkkijono on lyhyempi, kuin 255 merkkiä, täytetään loppu tyhjillä (CHR$(00)).
SVI DISK BASIC tallettaa FIELD-puskurin sisällön annetulle sektorille.

ESIM:

```
DSKO$ 0,23      ; kirjoittaa tiedon oletusaseman sektoriin 23. Toimii vain MSX
                  DISK BASIC:ssä.
DSKO$ 1,25,10   ; kirjoittaa FIELD-puskurin sisällön asemaan yksi, uralle 25 ja
                  sektoriin 10.
```

EOF

MUOTO: **EOF (X)**

VERSIOT: **MSX: KASET. +. LEVYKE**
SVI: KASET. + LEVYKE

TARKOITUS: Funktiolla testataan tiedoston loppumista INPUT-silmukan aikana.

LISÄYKSET: Ilmaisu X on luettavan tiedoston numero.
Mikäli EOF (End Of File) testausta EI tehdä ja tiedosto on luettu kokonaan ja annetaan INPUT #-käsky, tulostuu virheilmoitus "Input past end".
Funktio saa arvon —1 (TOSI), kun tiedosto loppuu.
EOF-funktiota tutkitaan yleensä IF-THEN -lauseella.

ESIM:

```
....
500 OPEN "CAS:KOE" FOR INPUT AS #1
510 IF EOF(1) THEN GOTO 550
520 INPUT #1,A%
530 PRINT A%
540 GOTO 510
550 CLOSE #1
....
```

Rivillä 510 tutkitaan tiedoston numero yksi (1) loppumista EOF-funktiolla ja arvolla —1 (TOSI) hypätään riville 550. Ohjelma pysyy silmukassa (rivit 520—540), kunnes EOF(1) toteutuu.

FIELD

MUOTO: **FIELD [#] <fnum>,<koko> AS <string> ,...**

VERSIOT: **MSX: LEVYKE**
SVI: LEVYKE

TARKOITUS: Käskyllä varataan tilaa annetuille merkkijonomuuttujille FIELD-puskurista.

LISÄYKSET: Käskyssä annettavien merkkijonomuuttujien yhteispituus ei saa ylittää 256 merkkiä.
Ilmaisu <fnum> on tiedostonumero, jolla tiedosto OPEN-käskyllä on avattu.
<koko> = merkkien määrä, joka varataan merkkijonomuuttujalle <string>.
Käskyllä voidaan varata samalla kertaa tilaa useammalle merkkijonomuuttujalle seuraavasti:

FIELD #1, 20 AS A$, 30 AS B$, 100 AS C$

Käsky varaa tilaa tiedoston yksi muuttujille A$ 20 merkkiä, B$ 30 merkkiä ja muuttujalle C$ 100 merkkiä.
FIELD-käsky EI sijoita puskuriin tietoa, vaan varaa muuttujille tilaa puskurista. Tiedon kirjoitus puskuriin tapahtuu LSET-, RSET- ja GET-käskyjen avulla.
Kun muuttujalle on varattu tilaa FIELD-puskurista, tulee muuttujalle sijoittaa tietoa vain LSET- ja RSET-käskyillä, joten INPUT #- ja PRINT #-käskyissä tulee käyttää eri muuttujanimiä!!
FIELD-käskyllä, määriteltävien muuttujien yhteispituus ei myöskään saa ylittää OPEN-käskyllä avatun tietueen pituutta (oletusarvo 256 tavua).
Yhtä avoinna olevaa tiedostoa kohti saa olla rajaton määrä FIELD-käskyjä voimassa (käskyt ovat voimassa koko tiedoston aukiolo ajan).

ESIM:

```
10  INPUT " ANNA NIMESI ";A$
20  INPUT " ANNA PUHELINNUMEROSI ";B
30  C$=MKD$(B)
40  OPEN "KOE" AS #1
50  FIELD #1,20 AS NIMI$,10 AS PUH$
60  LSET NIMI$=A$
70  LSET PUH$=C$
80  PUT#1,10
90  GET#1,10
100 B=CVD(PUH$)
110 PRINT NIMI$
120 PRINT B
130 CLOSE #1
```

Rivi 40 on kirjoitettava SVI-BASIC:ssä seuraavasti:

```
40  OPEN "1:KOE" AS #1
```

Riveillä 10 ja 20 syötetään nimi ja puhelinnumero. Rivillä 30 muunnetaan numero merkkijonoksi hajasaantitiedostoon kirjoituksen takia. Rivillä 40 avataan tiedosto nimellä "KOE", numerolla yksi (1). Rivillä 50 varataan FIELD-puskurista tilaa kahdelle merkkijonolle. Riveillä 60 ja 70 asetetaan tiedot puskuriin alkaen vasemmasta reunasta. Rivillä 80 kirjoitetaan puskurista tieto tiedostoon yksi ja rivillä 90 luetaan tieto takaisin puskuriin tiedostosta. Rivillä 100 suoritetaan muunnos takaisin numeeriseksi tiedoksi muuttujaan B. Riveillä 110 ja 120 tulostetaan luetut tiedot ja rivillä 130 suljetaan tiedosto.

FILES

MUOTO: **FILES [<määrittely>]**

VERSIOT: **MSX: LEVYKE**
SVI: LEVYKE

TARKOITUS: Käskyllä tulostetaan levykeasemassa olevien tiedostojen nimet kuvaruudulle.

LISÄYKSET: Jos ilmaisua <määrittely> ei anneta, tulostuu kaikkien ohjelmien nimet oletusarvoasemasta. Kysymysmerkillä (?) ja tähdellä (*) voidaan korvata joko yksi tai kaikki merkit (vain MSX DISK BASIC).
<määrittely> = MSX DISK BASIC:ssä voidaan antaa levykeasema kirjaimella A (asema yksi) tai B (asema kaksi), jota seuraa kaksoispiste (:) ja mahdollinen tarkennus. Tarkennuksessa voidaan pyytää vain jonkin tietyn ehdon täyttäviä tiedostonimiä, kuten esim. käsky FILES "A:*.DAT" tulostaisi kaikki ohjelmat, joiden tarkennusosa on "DAT".
SVI DISK BASIC:ssä ilmaisu <määrittely> voi sisältää levykeaseman tunnuksen 1 (asema yksi) tai 2 (asema kaksi).

ESIM:

KÄSKY	TULOSTAA
FILES	; kaikki ohjelmien ja tiedostojen nimet (oletusarvoasemasta).
FILES "A:*.*"	; kaikki ohjelmanimet asemasta A. Toimii vain MSX DISK BASIC:ssä
FILES 1	; kaikki ohjelmat asemasta yksi. Toimii vain SVI DISK BASIC:ssä
FILES "B:A*.*"	; kaikki A-kirjaimella alkavat ohjelmanimet asemasta B. Toimii vain MSX BASIC:ssä.

FPOS

MUOTO: **FPOS (X)**

VERSIOT: **MSX: ei käytössä**
SVI: LEVYKE

TARKOITUS: Funktio palauttaa tiedoston numero X fyysisen sijainnin levykkeellä (sektorin numeron).

LISÄYKSET: Tiedoston numero X tulee olla avoin (OPEN-käskyllä) ennen käskyn antamista.

ESIM:

```
10  OPEN "1:KOE" AS #1
20  PRINT FPOS(1)
30  CLOSE #1
40  KILL "1:KOE"
```

Ohjelma avaa tiedoston, tulostaa sen sijainnin levykkeellä, sulkee tiedoston ja lopuksi poistaa tiedoston levykkeeltä yksi.

GET

MUOTO: **GET [#] <fnum> [,<recnum>]**

VERSIOT: **MSX: LEVYKE**
SVI: LEVYKE

TARKOITUS: Käskyllä luetaan tietue hajasaantitiedostosta FIELD-puskuriin.

LISÄYKSET: <fnum> = tiedostonumero, jolla tiedosto on OPEN-käskyllä avattu.
<recnum> = tietueen numero, joka luetaan. Jos ilmaisu jätetään antamatta, luetaan seuraava (edellisen GET-käskyn jälkeen) tietue. Suurin mahdollinen käytettävä tietuenumero on MSX-BASIC:ssä 2^32-1 eli 4294967295 ja SVI-BASIC:ssä 32767.
GET-käskyn jälkeen voidaan käyttää INPUT #- ja LINE INPUT #-käskyä lukemaan merkit FIELD-puskurista.

ESIM:

```
....
200 OPEN "KOE" AS #1
210 FIELD #1, 20 AS A$, 10 AS B$
220 GET #1, 200
230 PRINT A$ ; B$
240 CLOSE #1
....
```

Rivillä 220 lukee GET-käsky tiedostosta yksi (#1) tietueen numerolla 200. Katso myös FIELD-käskyn esimerkkiä.

INPUT

MUOTO: **INPUT # <fnum>,<muuttuja> [,<muuttuja>,...]**

VERSIOT: **MSX: KASET. + LEVYKE**
SVI: KASET. + LEVYKE

TARKOITUS: Käskyllä luetaan sarjasaantitiedostosta tietoalkioita osoittaen ne muuttujien <muuttuja > sisällöksi.

LISÄYKSET: <fnum> = tiedostonumero, jolla tiedosto on OPEN-käskyllä avattu.
<muuttuja> = merkkijonomuuttuja tai lista muuttujista, jotka saavat sisällökseen luetut tiedot. Muuttujien tulee INPUT #-käskyssä vastata luettavien tietojen muotoa (merkkijonot on talletettava merkkijonomuuttujien sisällöksi ja numeeriset arvot numeerisiin muuttujiin).
INPUT #-käsky EI tulosta kysymysmerkkiä (?), kuten INPUT-käsky. Luettavat tiedot luetaan muuttujien listalla olevien muuttujien sisällöiksi, kuten INPUT-käskyllä. Numeerisiin arvoihin ei oteta huomioon tyhjiä (välilyöntejä), koodeja CHR$(13) eli ENTER, jne. Luvun katsotaan loppuvan ensimmäiseen tyhjään tai CHR$(13)-koodiin.
Kun tiedostoa luetaan INPUT #-käskyllä, saa ilmaisu <muuttuja> ensimmäisen merkin, kun saavutetaan ensimmäinen ei tyhjä, CHR$(13), tai CHR$(10). Jos ensimmäinen hyväksytty merkki on lainausmerkki ("), saa <muuttuja> kaikki merkit ensimmäisen ja seuraavan lainausmerkin väliltä. Tämän takia ei lainausmerkkiä voida taltioida osana merkkijonoa. Merkkijonon luku päättyy myös silloin, kun on luettu 255 merkkiä.

ESIM:

```
10  A%=1 : B%=2
20  OPEN "CAS:KOE" FOR OUTPUT AS #1
30  PRINT#1, A% , B%
40  CLOSE #1
50  INPUT "KELAA NAUHA ALKUUN JA PAINA JOTAIN NÄPPÄINTÄ"; A$
60  OPEN "CAS:KOE" FOR INPUT AS #1
70  INPUT#1, C% , D%
80  CLOSE #1
90  PRINT C% ; D%
```

Esimerkki sarjasaantitiedoston käytöstä. Rivillä 10 määritellään kahdelle muuttujalle arvo. Rivillä 20 avataan tiedosto nimellä "KOE" kirjoitusta varten. Rivillä 30 kirjoitetaan tiedostoon muuttujien A% ja B% sisällöt. Rivin 40 CLOSE-käsky sulkee tiedoston (nauhuri pysähtyy, jos REMOTE signaali on kytkettynä). Rivillä 50 annetaan mahdollisuus kelata nauha takaisin alkuun. Rivillä 60 avataan tiedosto lukua varten. Rivin 70 INPUT #-käsky lukee muuttujien C% ja D% tiedoiksi nauhalta kaksi lukuarvoa (HUOM! lukujen on oltava samaa tarkkuutta, kuin muuttujien C ja D eli kokonaislukuja). Rivillä 80 suljetaan tiedosto (nauhuri pysähtyy) ja rivillä 90 tulostetaan luetut luvut kuvaruudulle.

INPUT $

MUOTO: **INPUT$ (X [, [#] Y])**

VERSIOT:
MSX: KASET. + LEVYKE
SVI: KASET. + LEVYKE

TARKOITUS: Funktio palauttaa X-merkkiä pitkän merkkijonon, luettuna joko näppäimistöltä tai tiedostosta.

LISÄYKSET: Jos näppäimistöä käytetään syöttölaitteena, EI merkkejä kaiuteta (tulosteta) näytölle. Kaikki merkit, lukuunottamatta CTRL+C:tä ja CTRL+STOP:a (joilla sallitaan funktion keskeytys), sallitaan.
Ilmaisulla X annetaan määrä, joka merkkejä kerralla palautetaan ja ilmaisu Y on luettavan tiedoston numero.
Käskyllä voidaan mm. lukea levykkeeltä konekielitiedostoa ja tulostaa sen sisältö kuvaruudulle seuraavasti:

ESIM:

```
10  OPEN "A:SASA" FOR INPUT AS #1
20  IF EOF(1) THEN GOTO 50
30  PRINT HEX$(ASC(INPUT$(1,#1)));
40  GOTO 20
50  END
```

Rivi 10 on SVI-BASIC:ssä kirjoitettava seuraavasti:

```
10  OPEN "1:SASA" FOR INPUT AS #1
```

Rivillä 10 avataan tiedosto nimeltä "SASA" (konekielitiedosto, joka näkyy SVI-BASIC:ssä tiedostonimen edellä olevasta tähdestä, *) kirjoitusta varten. Rivillä 20 testataan tiedoston lopetusmerkin esiintymistä. Jos tiedosto on kokonaan luettu, saa EOF(1) arvon —1 eli TOSI ja suoritus siirtyy riville 50. Rivillä 30 tulostetaan heksadesimaalimuodossa luetun merkin (yksi kappale) ASCII-koodi. Riviltä 40 palataan riville 20 lukemaan seuraavaa merkkiä. Rivin 50 END-käsky päättää ohjelman.

IPL

MUOTO: **IPL "RUN" + CHR$(34) + "1:<fname>"**

VERSIOT:
MSX: ei käytössä
SVI: LEVYKE

TARKOITUS: Käskyllä voidaan asettaa ohjelmalle automaattikäynnistys.

LISÄYKSET: <fname> = ohjelman nimi, joka käynnistetään automaattisesti.
Levykkeen tulee olla asemassa kytkettäessä virta päälle (POWER ON).

ESIM:

```
10  PRINT "LEVYKKEELLÄ ON SEURAAVAT OHJELMAT:"
20  FILES
30  PRINT " ANNA OHJELMAN NIMI, JONKA"
```

```
40  PRINT "HALUAT KÄYNNISTYVÄN AUTOMAATTISESTI:"
50  INPUT A$
60  IF A$=" " THEN STOP
70  IPL "RUN"+CHR$ (&H22)+"1:"+A$
```

Ohjelma asettaa halutulle ohjelmalle automaattikäynnistyksen. Rivillä 10 tulostetaan tekstiä. Rivin 20 FILES-käsky tulostaa levykkeellä olevien ohjelmien nimet. Rivit 30 ja 40 tulostavat ohjeita ja rivillä 50 pyydetään syöttämään ohjelman nimi, jolle käynnistys halutaan asettaa. Rivillä 60 varmistetaan, onko syötetty tyhjä merkkijono ja jos on, pysäytetään ohjelman suoritus STOP-käskyllä. Rivillä 70 asetetaan annetulle ohjelmalle automaattikäynnistys. Huomaa, että CHR$ (34) on korvattu vastaavalla heksadesimaaliarvolla.

LFILES

MUOTO: **MSX: LFILES [<määrittely>]**

VERSIOT: **MSX: LEVYKE**
SVI: LEVYKE

TARKOITUS: Käskyllä tulostetaan levykkeellä olevien ohjelmien/tiedostojen nimet kirjoittimelle.

LISÄYKSET: <määrittely> = ilmaisulla voidaan antaa levykeasema, jolta ohjelmien nimet tulostetaan. Voidaan tulostaa myös tietyn ehdon täyttävät tiedostojen/ohjelmien nimet käyttämällä merkkejä "?" ja "*" (vain MSX DISK BASIC).
Käsky toimii samoin kuin FILES-käsky, mutta tulostus suoritetaan kirjoittimelle. Kts. FILES-käsky.

ESIM:

LFILES	; tulostaa kaikkien ohjelmien/tiedostojen nimet oletusarvoasemasta. Toimii sekä MSX- että SVI DISK BASIC:ssä.
LFILES 1	; tulostaa asemassa yksi olevien ohjelmien nimet. Toimii vain SVI DISK BASIC:ssä.
LFILES "B:*.BAS"	; tulostaa asemassa B (asema kaksi) olevien ohjelmien nimet, joiden tarkennusosa on BAS. Toimii vain MSX DISK BASIC:ssä.

LINE INPUT

MUOTO: **LINE INPUT # <fnum>,<string>**

VERSIOT: **MSX: KASET. + LEVYKE**
SVI: KASET. + LEVYKE

TARKOITUS: Käskyllä luetaan kokonainen rivi, ilman rajoituksia, sarjasaantitiedostosta merkkijonon <string> sisällöksi.

LISÄYKSET: <fnum> = tiedostonumero, jolla tiedosto on OPEN-käskyllä avattu.
Käskyllä voidaan lukea 254 merkkiä (maksimi).
<string> = merkkijonomuuttuja, joka saa sisällökseen luetut merkit. Kaikki merkit seuraavaan CR:n (CHR$ (13) eli ENTER) asti luetaan.
Käskyä käytetään mm. silloin, kun ASCII-tiedostoa halutaan hyödyntää toisella ohjelmalla. Tiedoston tulee olla talletettu ASCII-muodossa (kts. SAVE-käsky).
Käsky toimii, kuten INPUT #-käsky, mutta erona on kyky lukea kaikki merkit ENTER:iin asti (pilkut, pisteet, jne.).

ESIM:

```
10  LINE INPUT "ANNA JOKIN MERKKIJONO"; A$
20  OPEN "KOE" FOR OUTPUT AS #1
30  PRINT #1, A$
40  CLOSE #1
50  OPEN "KOE" FOR INPUT AS #1
60  LINE INPUT #1 , B$
70  CLOSE #1
80  PRINT B$
```

Ohjelma pyytää syöttämään rivillä 10 merkkijonoa, joka voi sisältää kaikkia merkkejä. Rivillä 20 avataan sarjasaantitiedosto levykkeelle kirjoitusta varten. Rivillä 30 kirjoitetaan syötetty merkkijono tiedostoon. Rivillä 40 suljetaan tiedosto ja rivillä 50 se avataan uudestaan lukua varten. Rivillä 60 luetaan kaikki merkit muuttujan B$ sisällöksi LINE INPUT # -käskyllä. Rivillä 70 suljetaan tiedosto. Rivillä 80 tulostetaan luettu merkkijono. Kokeile ohjelmaa syöttämällä erilaisia merkkijonoja. Rivi 20 ja 50 on SVI BASIC:ssä kirjoitettava seuraavasti:

```
20   OPEN "1:KOE" FOR OUTPUT AS #1
...
50   OPEN "1:KOE" FOR INPUT AS #1
```

LOC

MUOTO: LOC (<fnum>)

VERSIOT: MSX: LEVYKE
SVI: LEVYKE

TARKOITUS: Funktio palauttaa tietueen numeron, jolle avoinna olevan tiedoston numero <fnum> tiedot talletetaan.

LISÄYKSET: <fnum> = tiedostonumero, jolla tiedosto on OPEN-käskyllä avattu.
Hajasaantitiedostoilla LOC-funktio palauttaa tietueen numeron, johon GET- tai PUT-käsky on viimeksi kohdistunut. Jos tiedosto on avattu, mutta kirjoitusta/lukua tiedostoon EI ole ennen käskyn antamista suoritettu, palautuu nolla.
Sarjasaantitiedostoilla LOC-funktio palauttaa sektoreiden määrän (128 tavun palasten), joita on luettu/kirjoitettu tiedostoon OPEN-käskyn jälkeen.

ESIM:

```
....
510 IF LOC(1) < 1 THEN PRINT "EI TAPAHTUMIA TOISTAISEKSI." ELSE PRINT
"TIEDOSTOON ON KIRJOITETTU."
....
```

Rivillä 510 testataan, onko tiedostoon yksi (1) toistaiseksi kirjoitettu ja tulostetaan, onko vai ei.

LOF

MUOTO: LOF (<fnum>)

VERSIOT: MSX: LEVYKE
SVI: LEVYKE

TARKOITUS: Funktio palauttaa sektoreiden määrän.

LISÄYKSET: <fnum> = tiedostonumero, jonka käyttämien sektoreiden määrä halutaan tietää.
Jos tiedosto on avoin, mutta siihen ei ole kirjoitettu tai sitä ei ole luettu, palautuu nolla (0).

ESIM:

```
10   OPEN "KOE" FOR OUTPUT AS #1
20   A%=45
30   PRINT #1,A%
40   PRINT LOF(1)
50   CLOSE #1
```

Rivillä 10 avataan tiedosto nimellä "KOE" kirjoitusta varten. Rivillä 20 saa muuttuja A% arvon 45. Rivillä 30 kirjoitetaan muuttujan A% sisältö tiedostoon yksi (1). Rivillä 40 tulostetaan käytettyjen sektoreiden määrä. Rivin 50 CLOSE-käsky sulkee tiedoston. Rivi 10 on kirjoitettava SVI DISK BASIC:ssä seuraavasti:

```
10   OPEN "1:KOE" FOR OUTPUT AS #1
```

LSET

MUOTO: LSET <strvar> = <strexp>

VERSIOT: MSX: LEVYKE
SVI: LEVYKE

TARKOITUS: Käskyllä siirretään tietoa muistista (RAM) FIELD-puskuriin PUT-käskyä varten.

LISÄYKSET: <strvar> = merkkijonomuuttuja, jolle on FIELD-puskurista varattu tilaa.
<strexp> = merkkijonoilmaisu, joka talletetaan merkkijonomuuttujan <strvar> varaamaan tilaan. Talletus alkaa FIELD-puskurin vasemmasta reunasta. Vrt. RSET-käskyyn. Jos ilmaisu <strexp> sisältää vähemmän merkkejä, kuin ilmaisulla <strvar> on FIELD-puskuriin varattu, täytetään loput paikat puskurista nollilla (CHR$ (0)). Jos ilmaisu on pi-

dempi, kuin sille varattu tila, jätetään ylimääräiset merkit oikeasta laidasta huomioimatta. Numeeriset arvot on muunnettava ennen niiden tallentamista merkkijonoiksi (kts. MKI$-, MKS$- ja MKD$-käskyt.

ESIM:

```
10  CLEAR 600
20  INPUT "ANNA KAKSI MERKKIJONOA": A$ , B$
30  OPEN "KOE" AS #1
40  FIELD #1 , 10 AS EKA$, 20 AS TOKA$
50  LSET EKA$ = A$
60  LSET TOKA$ = B$
70  PUT #1 , 10
80  CLOSE #1
```

Rivi 30 on SVI-BASIC:ssä kirjoitettava seuraavasti:

```
30  OPEN "1:KOE" AS #1
```

Rivillä 10 varataan tilaa merkkijonoille. Rivillä 20 pyydetään syöttämään kaksi merkkijonoa. Rivillä 30 avataan tiedosto nimellä "KOE". Rivillä 40 varataan FIELD-puskurista tilaa merkkijonoille EKA$ ja TOKA$. Huomioi, että merkkijonoille varataan vain 10 ja 20 merkkiä tilaa, joten rivillä 20 syötetyistä merkkijonoista jätetään ylimääräiset pois. Riveillä 50 ja 60 kirjoitetaan tietoa FIELD-puskurin muuttujiin merkkijonoista A$ ja B$. Rivillä 70 viedään tieto PUT-käskyllä puskurista levykkeelle. Rivillä 80 suljetaan tiedosto yksi (1).

MAXFILES

MUOTO: **MAXFILES = <num>**

VERSIOT:
MSX: KASET. + LEVYKE
SVI: KASET. + LEVYKE

TARKOITUS: Käskyllä määritellään avoinna olevien tiedostojen maksimi määrä.

LISÄYKSET:
<num> = määrä, joka voi olla kokonaisluku välillä 0—15.
Jos ilmaisulla <num> annetaan arvo nolla (0), voidaan vain LOAD- ja SAVE-komentoja käyttää.
Ilmaisun <num> oletusarvo on yksi (1).

ESIM:

```
10  MAXFILES=5
20  OPEN "KOE1" FOR INPUT AS #1
30  OPEN "KOE2" FOR INPUT AS #2
40  OPEN "KOE3" FOR OUTPUT AS #3
....
```

Rivillä 10 asetetaan avoinna olevien tiedostojen maksimimääräksi viisi (5). Seuraavilla riveillä avataan tiedostoja lukua ja kirjoitusta varten.

MKI $, MKS $, MKD $

MUOTO:
MKI$ (<int>)
MKS$ (<int>)
MKD$ (<int>)

VERSIOT:
MSX: LEVYKE
SVI: LEVYKE

TARKOITUS: Funktiolla muunnetaan numeeriset arvot merkkijonoiksi.

LISÄYKSET:
Ennen lukujen taltioimista levykkeelle, on numeeriset arvot muunnettava merkkijonoiksi. MKI$-, MKS$- ja MKD$-funktiot muuntavat luvut merkkijonoiksi seuraavasti:
MKI$-funktio muuntaa kokonaisluvun kahden (2) tavun merkkijonoksi.
MKS$-funktio muuntaa yksinkertaisen tarkkuuden luvun neljän (4) tavun merkkijonoksi.
MKD$-funktio muuntaa kaksinkertaisen tarkkuuden luvun kahdeksan (8) tavun merkkijonoksi.
Funktioiden käänteisfunktiot ovat CVI, CVS ja CVD.
....

ESIM:

```
110  A# =SQR(7)
120  FIELD #1, 8 AS A$
130  LSET A$ = MKD$(A#)
140  PUT #1,10
....
```

Rivillä 110 saa kaksinkertaisen tarkkuuden muuttuja A# neliöjuuri seitsemän arvon. Rivillä 120 varataan FIELD-puskurista tilaa kahdeksan merkkiä merkkijonolle A$. Rivillä 130 asetetaan FIELD-puskuriin merkkijonolle A$ varattuun tilaan merkkijonoksi muuttujan A# sisältö. Rivin 140 PUT-käsky kirjoittaa tiedon levykkeelle tiedostoon yksi.

NAME

MUOTO: **NAME <old> AS <new>**

VERSIOT: **MSX: LEVYKE**
SVI: LEVYKE

TARKOITUS: Käskyllä vaihdetaan ohjelman/tiedoston nimi.

LISÄYKSET: <old> = vanha nimi, joka korvataan uudella.
<new> = uusi nimi, joka asettuu vanhan nimen paikalle.
Käskyllä voidaan vaihtaa BASIC-ohjelman, ASCII-tiedoston tai minkä tahansa tiedoston nimi. Käsky vaihtaa vain tiedoston nimen, eikä esim. muuta tiedoston sijaintia levykkeellä. Ilmaisun <old> niminen tiedoston/ohjelma tulee olla levykkeellä ja ilmaisun <new> nimistä tiedostoa EI ennen käskyn suoritusta saa levykkeellä olla.
Ilmaisujen <old> ja <new> muoto eroaa MSX- ja SVI-BASIC:ssä seuraavasti:

ESIM:

KÄSKY		VAIHTAA
NAME "1:KOE" AS "1:KOE.BAS"	;	levykkeellä yksi olevan ohjelman nimeltä "KOE" uudeksi nimeksi "KOE.BAS". Toimii vain SVI DISK BASIC:ssä.
NAME "KOE" AS "KOE.BAS"	;	sama kuin edellä, mutta käsky toimii vain MSX DISK BASIC:ssä. Käsky vaihtaa ohjelman oletusarvoasemasta.
NAME "A:KOE" AS "A:KOE.BAS"	;	sama kuin edellä, mutta nimi vaihdetaan levykeasemasta A (yksi). Käsky toimii vain MSX DISK BASIC:ssä.

OPEN

MUOTO: **OPEN "[<dev>] <nimi>" [FOR <mode>] AS [#] <fnum> [LEN=<n>]**

VERSIOT: **MSX: KASET. + LEVYKE**
SVI: KASET. + LEVYKE

TARKOITUS: Käskyllä avataan tiedosto tiedon lukua/kirjoitusta varten.

LISÄYKSET: <dev> = ilmaisulla annetaan I/O-laite, johon kirjoitus/luku kohdistuu. Voi olla jokin seuraavista:

DEV	VERSIOT		TARKOITTAA
A:	**MSX LEVYKE**	;	levykeasemaa A (yksi). Sekä luku että kirjoitus on mahdollinen.
B:	**MSX LEVYKE**	;	levykeasemaa B (kaksi). Sekä luku että kirjoitus on mahdollinen.
CAS:	**MSX ja SVI KASET.**	;	kasettitallenninta. Ilmaisu on oletusarvona MSX-BASIC:n kasettiversiossa ja SVI-BASIC:n kasetti- sekä levykeversioissa. Sekä luku että kirjoitus on mahdollinen.
1:	**SVI LEVYKE**	;	levykeasemaa yksi (1). Sekä luku että kirjoitus on mahdollinen.
2:	**SVI LEVYKE**	;	levykeasemaa kaksi (2). Sekä luku että kirjoitus on mahdollinen.
CRT:	**MSX ja SVI KASET. sekä LEVYKE**	;	kuvaruutua (tekstimoodia). Vain kirjoitus on mahdollinen.
GRP:	**MSX KASET. ja LEVYKE**	;	kuvaruutua (tarkkuusgrafiikkamoodia). Vain kirjoitus on mahdollinen.

KYBD:	**SVI KASET. ja LEVY-KE**	; näppäimistöä. Vain luku on mahdollinen.
LPT:	**MSX ja SVI KASET. ja LEVYKE**	; kirjoitinta. Vain kirjoitus on mahdollinen.

Jos <dev> jätetään antamatta, olettaa SVI-BASIC sekä levyke- että kasettiversiossa laitteen olevan kasettinauhurin. MSX-BASIC olettaa kasettiversiossa laitteen olevan kasettitallentimen ja levykeversiossa oletusarvoaseman.
<nimi> = avattavan tiedoston nimi. MSX-BASIC:ssä tiedostonimi voi sisältää kahdeksan (8) merkkiä ja kolme (3) merkkiä tarkennusosassa. SVI-BASIC:ssä tiedostonimi voi sisältää kuusi (6) merkkiä varsinaisessa nimessä ja kolme (3) merkkiä tarkennusosassa.
<nimi> = moodi on jokin seuraavista:

MOODI		**TARKOITTAA**
FOR APPEND	;	lisäysmoodi. Tiedoston perään lisätään tietoa avattavalla tiedostolla.
FOR INPUT	;	lukumoodi. Tiedosto avataan tiedon lukua varten annetulta laitteelta.
FOR OUTPUT	;	kirjoitusmoodi. Tiedosto avataan tiedon muistista annetulle laitteelle kirjoitusta varten.

Ilmaisu <mode> oletusarvona on luku/kirjoitusmoodi eli molemmat moodit ovat mahdollisia.
<fnum> = avattavan tiedoston numero. Ilmaisun arvo tulee olla välillä 1—15 ja pienempi, kuin MAXFILES-käskyllä annettu suurin tiedostonumero. Tiedostonumero on käytössä niin kauan, kunnes CLOSE-käskyllä tiedosto suljetaan.
<ln> = ilmaisulla voidaan määritellä käytettävien tietueiden maksimi pituus. Oletusarvona on 256 tavua (merkkiä). Ilmaisu voi saada arvokseen kokonaisluvun väliltä 1—256. Ilmaisua voidaan käyttää vain MSX DISK BASIC:ssä.
Jos sarjasaantitiedostoa käytettäessä moodissa APPEND tai INPUT ei tiedostoa annetulla nimellä löydy, tulostuu virheilmoitus "File not found" ja jos OUTPUT-moodia käytettäessä annetaan jo olemassa oleva tiedostonimi, poistuu vanha tiedosto.
Sarjasaantitiedosto (tai hajasaantitiedosto) voidaan avata OPEN-käskyllä lukua varten useammalla, kuin yhdellä numerolla kerrallaan. Kirjoitusta varten voidaan tiedosto avata kerrallaan vain yhdellä numerolla.

ESIM: **OPEN "A:KOE" FOR OUTPUT AS #1 LEN=128**

Käsky avaa tiedoston nimellä "KOE" levykeasemaan A kirjoitusta varten numerolla yksi (1) ja asettaa samalla käytettävien tietueiden pituuksiksi 128 tavua. Käsky toimii vain MSX DISK BASIC:ssä.
Avattavan tiedoston nimi voidaan syöttää ensin INPUT-lauseella ja sijoittaa annettu merkkijono OPEN-lauseeseen.

```
....
120  INPUT "ANNA AVATTAVAN TIEDOSTON NIMI"; A$
130  A$="A:"+A$
140  OPEN A$ FOR INPUT AS #1
....
```

Tiedostonimeen lisätään merkintä siitä, mille laitteelle tietoa halutaan tallentaa (rivi 130). Vastaavasti voidaan syöttää myös avattavan tiedoston numero. Rivi 130 on annettava SVI-BASIC:ssä seuraavasti (jos halutaan sama toiminta):

```
130  A$="1:"+A$
```

Seuraavaksi esimerkki tiedostoon kirjoituksesta ja luvusta, jossa kirjoitetaan kaksi tietoa sarjasaantitiedostoon, suljetaan tiedosto, avataan uudelleen lukua varten, luetaan tiedot ja lopuksi tulostetaan ne näytölle.

```
10   A$="TIETO1" : B$="TIETO2" : C$=","
20   OPEN "CAS:KOE" FOR OUTPUT AS 1
30   PRINT #1, A$ , C$ , B$
40   CLOSE 1
50   INPUT "KELAA NAUHURI ALKUUN JA PAINA JOTAIN NÄPPÄINTÄ" ;
     A$
60   OPEN "CAS:KOE" FOR INPUT AS 1
70   INPUT #1, A$ , B$
80   CLOSE 1
90   PRINT A$
100  PRINT B$
```

Huomaa, että jos merkkijonon C$ sisältämää pilkkua EI lisättäisi muuttujien A$ ja B$ väliin, tallentuisi tieto yhtenä kappaleena "TIETO1TIETO2". Tällöin myös rivillä 70 oleva INPUT#-lause antaisi virheilmoituksen (komento siirtyisi komentotilaan) "Input past end", joka ilmoittaisi, että tiedoston lopetusmerkki on tullut, vaikka tietoa ollaan oltu vielä lukemassa.

PRINT

MUOTO: **PRINT # <fnum>,<string> [,<string2> ,...]**

VERSIOT: **MSX: KASET. + LEVYKE**
SVI: KASET. + LEVYKE

TARKOITUS: Käskyllä kirjoitetaan tietoa avoinna olevaan tiedostoon.

LISÄYKSET: <fnum> = tiedostonumero, johon tietoa kirjoitetaan. On kokonaisluku välillä 1—15.
<string> = merkkijonomuuttuja, merkkijono, numero tai numeerinen muuttuja, joka/jotka tiedostoon kirjoitetaan.
Yhdellä PRINT#-käskyllä voidaan kirjoittaa useampia tietueita. Kirjoitus PRINT#-käskyllä kohdistuu aina sarjasaantitiedostoon.
Jos tiedot halutaan kirjoittaa tiedostoon peräkkäin ilman eroittimia (tämä tulee ottaa huomioon myös lukuvaiheessa), erotetaan ilmaisut <string> toisistaan puolipisteillä (;).

PRINT #1, A;B;C!;FF#;S%

Jos kirjoitettavat tiedot (merkkijonot) sisältävät itsessään pilkkuja, puolipisteitä, edeltäviä merkitseviä välilyöntejä (tyhjiä) tai CHR$(13)-koodeja (eli ENTER:ä vastaava koodi), tulee tiedot erotella lainausmerkeillä eli CHR$(34)-koodilla.

ESIM: **PRINT #1,CHR$(34);"TIETO,ASIA";CHR$(34)**

Käsky kirjoittaa tiedon "TIETO,ASIA" tiedostoon yksi (1).

PRINT # USING

MUOTO: **PRINT # <fnum>,[USING "<muoto>" ;] <string> [,<string2>,...]**

VERSIOT: **MSX: KASET. + LEVYKE**
SVI: KASET. + LEVYKE

TARKOITUS: Käskyllä kirjoitetaan tietoa tiedostoon numero <fnum> annetulla muodolla <muoto>.

LISÄYKSET: <fnum> = tiedoston numero, johon tieto kirjoitetaan. Kokonaisluku väliltä 1—15.
<muoto> = muoto, jolla tiedot talletetaan. Muoto annetaan samoin, kuin PRINT USING -lauseella.
<string> = merkkijono tai numeerinen arvo, joka tiedostoon talletetaan.
Kts. PRINT USING -käsky.

ESIM: **PRINT #1 USING "###.##"; A , B**

Käsky kirjoittaa sarjasaantiedostoon numero yksi (1) annetulla muodolla (viidellä numerolla, joista kaksi desimaaliosassa) numeeriset muuttujat A:n ja B:n.

PUT

MUOTO: **PUT [#] <fnum> [,<recnum>]**

VERSIOT: **MSX: LEVYKE**
SVI: LEVYKE

TARKOITUS: Käskyllä kirjoitetaan tietue FIELD-puskurista hajasaantitiedostoon numero <fnum>.

LISÄYKSET: <fnum> = tiedoston numero, jolla tiedosto on OPEN-käskyllä avattu. Kokonaisluku välillä 1—15.
<recnum> = tietueen numero, jolla tietue kirjoitetaan. Jos ilmaisua EI määritellä, käytetään seuraavaa mahdollista numeroa edellisen PUT-käskyn jälkeen. Tietuenumerot voivat olla seuraavia:

MSX-BASIC: 1—4294967295
SVI-BASIC: 1—32767

Ennen PUT-käskyn antamista, täytyy tietue siirtää FIELD-puskuriin LSET- tai RSET-käskyllä.

ESIM:

```
10  INPUT "ANNA ETUNIMESI" ; A$
20  INPUT "ANNA SUKUNIMESI" ; B$
30  OPEN "A:KOE" AS #1
40  FIELD #1, 10 AS C$ , 20 AS D$
50  LSET C$ = A$
60  LSET D$ = B$
70  PUT #1, 500
80  CLOSE #1
```

Ohjelma kirjoittaa annetut nimet tiedostoon nimeltä "KOE". Rivi 30 on SVI BASIC:ssä kirjoitettava seuraavasti:

```
30  OPEN "1:KOE" AS #1
```

Rivillä 10 ja 20 pyydetään käyttäjää syöttämään nimet. Rivillä 30 avataan tiedosto nimellä "KOE". Rivillä 40 varataan FIELD-puskurista tilaa kahdelle merkkijonolle. Rivillä 50 kirjoitetaan merkkijono A$ puskuriin merkkijonomuuttujan C$ varaamaan tilaan. Rivillä 60 vastaavasti merkkijonomuuttujan D$ varaamaan tilaan kirjoitetaan merkkijonon B$ sisältö. Rivillä 70 kirjoitetaan puskurin sisältö tiedostoon yksi (1), tietueeseen numero 500. Rivin 80 CLOSE-käsky sulkee tiedoston.

RSET

MUOTO: **RSET <strvar> = <strexp>**

VERSIOT: **MSX: LEVYKE**
SVI: LEVYKE

TARKOITUS: Käskyllä siirretään tietoa muistista (RAM) FIELD-puskuriin PUT-käskyä varten.

LISÄYKSET: <strvar> = merkkijonomuuttuja, jolle on FIELD-puskurista varattu tilaa.
<strexp> = merkkijonoilmaisu, joka talletetaan merkkijonomuuttujan <strvar> varaamaan tilaan. Talletus alkaa FIELD-puskurin oikeasta reunasta. Vrt. LSET-käskyyn.
Jos ilmaisu <strexp> sisältää vähemmän merkkejä, kuin ilmaisulla <strvar> on FIELD-puskuriin varattu, täytetään loput paikat puskurista nollilla (CHR$ (0)). Jos ilmaisu on pidempi, kuin sille varattu tila, jätetään ylimääräiset merkit vasemmasta laidasta huomioimatta.
Numeeriset arvot on muunnettava ennen niiden tallentamista merkkijonoiksi (kts. MKI$-, MKS$- ja MKD$-käskyt).

ESIM:

```
10  CLEAR 600
20  INPUT "ANNA KAKSI MERKKIJONOA"; A$ , B$
30  OPEN "KOE" AS #1
40  FIELD #1 , 10 AS EKA$ , 20 AS TOKA$
50  RSET EKA$ = A$
60  RSET TOKA$ = B$
70  PUT #1 , 10
80  CLOSE #1
```

Rivi 30 on SVI-BASIC:ssä kirjoitettava seuraavasti:

```
30  OPEN "1:KOE" AS #1
```

Rivillä 10 varataan tilaa merkkijonoille. Rivillä 20 pyydetään syöttämään kaksi merkkijonoa. Rivillä 30 avataan tiedosto nimellä "KOE". Rivillä 40 varataan FIELD-puskurista tilaa merkkijonoille EKA$ ja TOKA$. Huomioi, että merkkijonoille varataan vain 10 ja 20 merkkiä tilaa, joten rivillä 20 syötetyistä merkkijonoista jätetään ylimääräiset pois. Riveillä 50 ja 60 kirjoitetaan tietoa FIELD-puskurin muuttujiin merkkijonoista A$ ja B$. Rivillä 70 viedään tieto PUT-käskyllä puskurista levykkeelle. Rivillä 80 suljetaan tiedosto yksi (1).

SET

MUOTO: **SET <drive>,<attr>**
SET "<drv>:<name>",<attr>
SET #<fnum>,<attr>

VERSIOT: **MSX: ei käytössä**
SVI: LEVYKE

TARKOITUS: Käskyllä asetetaan ohjelmalle/tiedostolle ehto.

LISÄYKSET: <drive> = ilmaisulla määritellään levykeasema, jonka tila asetetaan ilmaisulla <attr> määriteltyyn tilaan. Voi olla yksi (1), jolla tarkoitetaan levykeasemaa yksi tai kaksi (2), jolla tarkoitetaan vastaavasti asemaa kaksi.
<drv> = sama, kuin edellä.
<name> = ohjelman/tiedoston nimi, jolle ehto annetaan. Tulee vastata täydellisesti levykkeellä olevaa tiedoston/ohjelman nimeä.
<fnum> = tiedoston numero, jolle ehto annetaan. Numero on OPEN-käskyllä määritelty tiedoston numero (tiedoston tulee olla siis avoin).
<attr> = ehto, joka tiedostolle/ohjelmalle annetaan. Voi olla jokin seuraavista:

P	;	kirjoitussuojaus. Suojaus on voimassa, kunnes toisin määritellään. Voidaan antaa koko levykkeelle (SET <drv>, P).
R	;	pakottaa koneen lukemaan tiedoston/ohjelman jokaisen kirjoitusoperaation jälkeen.
""	;	poistaa voimassa olevan ehdon.

ESIM:

SET "1:KOE",P	;	asettaa ohjelmalle/tiedostolle nimeltä "KOE" kirjoitussuojauksen.
SET 1,P	;	asettaa levykeasemassa yksi (1) olevalle levykkeelle kirjoitussuojauksen.
SET #1,P	;	asettaa avoinna olevalle tiedostolle numero yksi (1) kirjoitussuojauksen.
SET 2,""	;	poistaa levykeasemassa kaksi (2) olevasta levykkeestä ehdot.

VARPTR

MUOTO: **VARPTR (#<fnum>)**

VERSIOT: **MSX: LEVYKE**
SVI: LEVYKE

TARKOITUS: Funktio palauttaa tiedoston numero <fnum> sijainnin muistissa.

LISÄYKSET: <fnum> = tiedostonumero, jolla tiedosto on OPEN-käskyllä avattu.
Palautettavaan lukuun on lisättävä arvo 65536, jotta saataisiin oikea osoite.
Funktion avulla voidaan tutkia "tiedostoplokkejen" sijaintia RAM-muistissa.

ESIM:

```
10  OPEN "A:KOE" FOR OUTPUT AS 1
20  PRINT VARPTR (#1) + 65536
30  CLOSE
```

Ohjelma tulostaa tiedoston numero yksi (1) sijainnin muistissa.